



OpenShift Container Platform 4.12

Installing

Installing and configuring OpenShift Container Platform clusters

OpenShift Container Platform 4.12 Installing

Installing and configuring OpenShift Container Platform clusters

Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document provides information about installing OpenShift Container Platform and details about some configuration processes.

Table of Contents

CHAPTER 1. OPENSIFT CONTAINER PLATFORM INSTALLATION OVERVIEW	76
1.1. ABOUT OPENSIFT CONTAINER PLATFORM INSTALLATION	76
1.1.1. About the installation program	76
1.1.2. About Red Hat Enterprise Linux CoreOS (RHCOS)	77
1.1.3. Glossary of common terms for OpenShift Container Platform installing	77
1.1.4. Installation process	79
The installation process with the Assisted Installer	80
The installation process with Agent-based infrastructure	80
The installation process with installer-provisioned infrastructure	80
The installation process with user-provisioned infrastructure	81
Installation process details	81
1.1.5. Verifying node state after installation	83
Installation scope	84
1.1.6. OpenShift Local overview	84
1.2. SUPPORTED PLATFORMS FOR OPENSIFT CONTAINER PLATFORM CLUSTERS	85
CHAPTER 2. SELECTING A CLUSTER INSTALLATION METHOD AND PREPARING IT FOR USERS	87
2.1. SELECTING A CLUSTER INSTALLATION TYPE	87
2.1.1. Do you want to install and manage an OpenShift Container Platform cluster yourself?	87
2.1.2. Have you used OpenShift Container Platform 3 and want to use OpenShift Container Platform 4?	88
2.1.3. Do you want to use existing components in your cluster?	88
2.1.4. Do you need extra security for your cluster?	89
2.2. PREPARING YOUR CLUSTER FOR USERS AFTER INSTALLATION	89
2.3. PREPARING YOUR CLUSTER FOR WORKLOADS	90
2.4. SUPPORTED INSTALLATION METHODS FOR DIFFERENT PLATFORMS	90
CHAPTER 3. CLUSTER CAPABILITIES	96
3.1. SELECTING CLUSTER CAPABILITIES	96
3.2. OPTIONAL CLUSTER CAPABILITIES IN OPENSIFT CONTAINER PLATFORM 4.12	97
3.2.1. Bare-metal capability	97
Purpose	98
3.2.2. Cluster storage capability	98
Purpose	98
Notes	98
3.2.3. Console capability	98
Purpose	98
3.2.4. CSI snapshot controller capability	99
Purpose	99
3.2.5. Insights capability	99
Purpose	99
Notes	99
3.2.6. Marketplace capability	99
Purpose	99
3.2.7. OpenShift samples capability	100
Purpose	100
3.3. ADDITIONAL RESOURCES	100
CHAPTER 4. DISCONNECTED INSTALLATION MIRRORING	101
4.1. ABOUT DISCONNECTED INSTALLATION MIRRORING	101
4.1.1. Creating a mirror registry	101
4.1.2. Mirroring images for a disconnected installation	101
4.2. CREATING A MIRROR REGISTRY WITH MIRROR REGISTRY FOR RED HAT OPENSIFT	101

4.2.1. Prerequisites	101
4.2.2. Mirror registry for Red Hat OpenShift introduction	102
4.2.2.1. Mirror registry for Red Hat OpenShift limitations	102
4.2.3. Mirroring on a local host with mirror registry for Red Hat OpenShift	103
4.2.4. Updating mirror registry for Red Hat OpenShift from a local host	104
4.2.5. Mirroring on a remote host with mirror registry for Red Hat OpenShift	105
4.2.6. Updating mirror registry for Red Hat OpenShift from a remote host	106
4.2.7. Replacing mirror registry for Red Hat OpenShift SSL/TLS certificates	106
4.2.8. Uninstalling the mirror registry for Red Hat OpenShift	107
4.2.9. Mirror registry for Red Hat OpenShift flags	108
4.2.10. Mirror registry for Red Hat OpenShift release notes	109
4.2.10.1. Mirror registry for Red Hat OpenShift 1.3.11	109
4.2.10.2. Mirror registry for Red Hat OpenShift 1.3.10	109
4.2.10.3. Mirror registry for Red Hat OpenShift 1.3.9	110
4.2.10.4. Mirror registry for Red Hat OpenShift 1.3.8	110
4.2.10.5. Mirror registry for Red Hat OpenShift 1.3.7	110
4.2.10.6. Mirror registry for Red Hat OpenShift 1.3.6	110
4.2.10.7. Mirror registry for Red Hat OpenShift 1.3.5	110
4.2.10.8. Mirror registry for Red Hat OpenShift 1.3.4	111
4.2.10.9. Mirror registry for Red Hat OpenShift 1.3.3	111
4.2.10.10. Mirror registry for Red Hat OpenShift 1.3.2	111
4.2.10.11. Mirror registry for Red Hat OpenShift 1.3.1	111
4.2.10.12. Mirror registry for Red Hat OpenShift 1.3.0	111
4.2.10.12.1. New features	111
4.2.10.12.2. Bug fixes	112
4.2.10.13. Mirror registry for Red Hat OpenShift 1.2.9	112
4.2.10.14. Mirror registry for Red Hat OpenShift 1.2.8	112
4.2.10.15. Mirror registry for Red Hat OpenShift 1.2.7	112
4.2.10.15.1. Bug fixes	113
4.2.10.16. Mirror registry for Red Hat OpenShift 1.2.6	113
4.2.10.16.1. New features	113
4.2.10.17. Mirror registry for Red Hat OpenShift 1.2.5	113
4.2.10.18. Mirror registry for Red Hat OpenShift 1.2.4	113
4.2.10.19. Mirror registry for Red Hat OpenShift 1.2.3	113
4.2.10.20. Mirror registry for Red Hat OpenShift 1.2.2	113
4.2.10.21. Mirror registry for Red Hat OpenShift 1.2.1	113
4.2.10.22. Mirror registry for Red Hat OpenShift 1.2.0	114
4.2.10.22.1. Bug fixes	114
4.2.10.23. Mirror registry for Red Hat OpenShift 1.1.0	114
4.2.10.23.1. New features	114
4.2.10.23.2. Bug fixes	114
4.2.11. Troubleshooting mirror registry for Red Hat OpenShift	114
4.2.12. Additional resources	115
4.3. MIRRORING IMAGES FOR A DISCONNECTED INSTALLATION	115
4.3.1. Prerequisites	116
4.3.2. About the mirror registry	116
4.3.3. Preparing your mirror host	117
4.3.3.1. Installing the OpenShift CLI by downloading the binary	117
Installing the OpenShift CLI on Linux	117
Installing the OpenShift CLI on Windows	118
Installing the OpenShift CLI on macOS	118
4.3.4. Configuring credentials that allow images to be mirrored	119
4.3.5. Mirroring the OpenShift Container Platform image repository	121

4.3.6. The Cluster Samples Operator in a disconnected environment	124
4.3.6.1. Cluster Samples Operator assistance for mirroring	124
4.3.7. Mirroring Operator catalogs for use with disconnected clusters	125
4.3.7.1. Prerequisites	126
4.3.7.2. Extracting and mirroring catalog contents	126
4.3.7.2.1. Mirroring catalog contents to registries on the same network	126
4.3.7.2.2. Mirroring catalog contents to airgapped registries	128
4.3.7.3. Generated manifests	130
4.3.7.4. Postinstallation requirements	131
4.3.8. Next steps	131
4.3.9. Additional resources	132
4.4. MIRRORING IMAGES FOR A DISCONNECTED INSTALLATION USING THE OC-MIRROR PLUGIN	132
4.4.1. About the oc-mirror plugin	132
4.4.2. oc-mirror compatibility and support	133
4.4.3. About the mirror registry	133
4.4.4. Prerequisites	134
4.4.5. Preparing your mirror hosts	134
4.4.5.1. Installing the oc-mirror OpenShift CLI plugin	134
4.4.5.2. Configuring credentials that allow images to be mirrored	135
4.4.6. Creating the image set configuration	138
4.4.7. Mirroring an image set to a mirror registry	140
4.4.7.1. Mirroring an image set in a partially disconnected environment	140
4.4.7.1.1. Mirroring from mirror to mirror	140
4.4.7.1.2. Mirroring from mirror to disk	141
4.4.7.2.1. Mirroring from disk to mirror	143
4.4.7.2. Mirroring an image set in a fully disconnected environment	141
4.4.7.2.1. Mirroring from mirror to disk	141
4.4.7.2.2. Mirroring from disk to mirror	143
4.4.8. Configuring your cluster to use the resources generated by oc-mirror	143
4.4.9. Keeping your mirror registry content updated	144
4.4.9.1. About updating your mirror registry content	145
Adding new and updated images	145
Pruning images	145
4.4.9.2. Updating your mirror registry content	146
4.4.10. Performing a dry run	147
4.4.11. Mirroring file-based catalog Operator images in OCI format	148
4.4.12. Image set configuration parameters	151
4.4.13. Image set configuration examples	157
Use case: Including the shortest OpenShift Container Platform upgrade path	158
Use case: Including all versions of OpenShift Container Platform from a minimum to the latest	158
Use case: Including Operator versions from a minimum to the latest	158
Use case: Including the Nutanix CSI Operator	159
Use case: Including the default Operator channel	160
Use case: Including an entire catalog (all versions)	160
Use case: Including an entire catalog (channel heads only)	161
Use case: Including arbitrary images and helm charts	161
4.4.14. Command reference for oc-mirror	161
4.4.15. Additional resources	163
CHAPTER 5. INSTALLING ON ALIBABA	164
5.1. PREPARING TO INSTALL ON ALIBABA CLOUD	164
5.1.1. Prerequisites	164
5.1.2. Requirements for installing OpenShift Container Platform on Alibaba Cloud	164
5.1.3. Registering and Configuring Alibaba Cloud Domain	164
5.1.4. Supported Alibaba regions	165

5.1.5. Next steps	165
5.2. CREATING THE REQUIRED ALIBABA CLOUD RESOURCES	165
5.2.1. Creating the required RAM user	165
5.2.2. Configuring the Cloud Credential Operator utility	170
5.2.3. Next steps	171
5.3. INSTALLING A CLUSTER QUICKLY ON ALIBABA CLOUD	171
5.3.1. Prerequisites	172
5.3.2. Internet access for OpenShift Container Platform	172
5.3.3. Generating a key pair for cluster node SSH access	172
5.3.4. Obtaining the installation program	174
5.3.5. Creating the installation configuration file	175
5.3.6. Generating the required installation manifests	177
5.3.7. Creating credentials for OpenShift Container Platform components with the ccoctl tool	177
5.3.8. Deploying the cluster	180
5.3.9. Installing the OpenShift CLI by downloading the binary	181
Installing the OpenShift CLI on Linux	181
Installing the OpenShift CLI on Windows	182
Installing the OpenShift CLI on macOS	182
5.3.10. Logging in to the cluster by using the CLI	183
5.3.11. Logging in to the cluster by using the web console	183
5.3.12. Telemetry access for OpenShift Container Platform	184
5.3.13. Next steps	185
5.4. INSTALLING A CLUSTER ON ALIBABA CLOUD WITH CUSTOMIZATIONS	185
5.4.1. Prerequisites	185
5.4.2. Internet access for OpenShift Container Platform	185
5.4.3. Generating a key pair for cluster node SSH access	186
5.4.4. Obtaining the installation program	188
5.4.4.1. Creating the installation configuration file	188
5.4.4.2. Generating the required installation manifests	190
5.4.4.3. Creating credentials for OpenShift Container Platform components with the ccoctl tool	190
5.4.4.4. Installation configuration parameters	193
5.4.4.4.1. Required configuration parameters	193
5.4.4.4.2. Network configuration parameters	195
5.4.4.4.3. Optional configuration parameters	197
5.4.4.4.4. Additional Alibaba Cloud configuration parameters	202
5.4.4.5. Sample customized install-config.yaml file for Alibaba Cloud	205
5.4.4.6. Configuring the cluster-wide proxy during installation	206
5.4.5. Deploying the cluster	208
5.4.6. Installing the OpenShift CLI by downloading the binary	210
Installing the OpenShift CLI on Linux	210
Installing the OpenShift CLI on Windows	211
Installing the OpenShift CLI on macOS	211
5.4.7. Logging in to the cluster by using the CLI	212
5.4.8. Logging in to the cluster by using the web console	212
5.4.9. Telemetry access for OpenShift Container Platform	213
5.4.10. Next steps	213
5.5. INSTALLING A CLUSTER ON ALIBABA CLOUD WITH NETWORK CUSTOMIZATIONS	214
5.5.1. Prerequisites	214
5.5.2. Internet access for OpenShift Container Platform	214
5.5.3. Generating a key pair for cluster node SSH access	215
5.5.4. Obtaining the installation program	216
5.5.5. Network configuration phases	217
5.5.5.1. Creating the installation configuration file	218

5.5.5.2. Generating the required installation manifests	219
5.5.5.3. Installation configuration parameters	220
5.5.5.3.1. Required configuration parameters	220
5.5.5.3.2. Network configuration parameters	222
5.5.5.3.3. Optional configuration parameters	224
5.5.5.4. Sample customized install-config.yaml file for Alibaba Cloud	229
5.5.5.5. Configuring the cluster-wide proxy during installation	230
5.5.6. Cluster Network Operator configuration	232
5.5.6.1. Cluster Network Operator configuration object	232
defaultNetwork object configuration	233
Configuration for the OpenShift SDN network plugin	234
Configuration for the OVN-Kubernetes network plugin	235
kubeProxyConfig object configuration	239
5.5.7. Specifying advanced network configuration	240
5.5.8. Configuring hybrid networking with OVN-Kubernetes	241
5.5.9. Deploying the cluster	243
5.5.10. Installing the OpenShift CLI by downloading the binary	244
Installing the OpenShift CLI on Linux	244
Installing the OpenShift CLI on Windows	245
Installing the OpenShift CLI on macOS	245
5.5.11. Logging in to the cluster by using the CLI	246
5.5.12. Logging in to the cluster by using the web console	247
5.5.13. Telemetry access for OpenShift Container Platform	247
5.5.14. Next steps	248
5.6. INSTALLING A CLUSTER ON ALIBABA CLOUD INTO AN EXISTING VPC	248
5.6.1. Prerequisites	248
5.6.2. Using a custom VPC	249
5.6.2.1. Requirements for using your VPC	249
5.6.2.2. VPC validation	249
5.6.2.3. Division of permissions	249
5.6.2.4. Isolation between clusters	250
5.6.3. Internet access for OpenShift Container Platform	250
5.6.4. Generating a key pair for cluster node SSH access	250
5.6.5. Obtaining the installation program	252
5.6.5.1. Creating the installation configuration file	253
5.6.5.2. Installation configuration parameters	254
5.6.5.2.1. Required configuration parameters	255
5.6.5.2.2. Network configuration parameters	256
5.6.5.2.3. Optional configuration parameters	258
5.6.5.2.4. Additional Alibaba Cloud configuration parameters	263
5.6.5.3. Sample customized install-config.yaml file for Alibaba Cloud	266
5.6.5.4. Generating the required installation manifests	267
5.6.5.5. Configuring the Cloud Credential Operator utility	268
5.6.5.6. Creating credentials for OpenShift Container Platform components with the ccoctl tool	269
5.6.6. Deploying the cluster	272
5.6.7. Installing the OpenShift CLI by downloading the binary	273
Installing the OpenShift CLI on Linux	274
Installing the OpenShift CLI on Windows	274
Installing the OpenShift CLI on macOS	275
5.6.8. Logging in to the cluster by using the CLI	275
5.6.9. Logging in to the cluster by using the web console	276
5.6.10. Telemetry access for OpenShift Container Platform	277
5.6.11. Next steps	277

5.7. UNINSTALLING A CLUSTER ON ALIBABA CLOUD	277
5.7.1. Removing a cluster that uses installer-provisioned infrastructure	277
CHAPTER 6. INSTALLING ON AWS	279
6.1. PREPARING TO INSTALL ON AWS	279
6.1.1. Prerequisites	279
6.1.2. Requirements for installing OpenShift Container Platform on AWS	279
6.1.3. Choosing a method to install OpenShift Container Platform on AWS	279
6.1.3.1. Installing a cluster on installer-provisioned infrastructure	279
6.1.3.2. Installing a cluster on user-provisioned infrastructure	280
6.1.4. Next steps	280
6.2. CONFIGURING AN AWS ACCOUNT	280
6.2.1. Configuring Route 53	280
6.2.1.1. Ingress Operator endpoint configuration for AWS Route 53	281
6.2.2. AWS account limits	282
6.2.3. Required AWS permissions for the IAM user	284
6.2.4. Creating an IAM user	292
6.2.5. IAM Policies and AWS authentication	293
6.2.5.1. Default permissions for IAM instance profiles	293
6.2.5.2. Specifying an existing IAM role	295
6.2.5.3. Using AWS IAM Analyzer to create policy templates	296
6.2.6. Supported AWS Marketplace regions	296
6.2.7. Supported AWS regions	297
6.2.7.1. AWS public regions	297
6.2.7.2. AWS GovCloud regions	298
6.2.7.3. AWS SC2S and C2S secret regions	298
6.2.7.4. AWS China regions	298
6.2.8. Next steps	298
6.3. MANUALLY CREATING IAM FOR AWS	299
6.3.1. Alternatives to storing administrator-level secrets in the kube-system project	299
6.3.2. Manually create IAM	300
6.3.3. Mint mode	303
6.3.4. Mint mode with removal or rotation of the administrator-level credential	303
6.3.5. Next steps	304
6.4. INSTALLING A CLUSTER QUICKLY ON AWS	304
6.4.1. Prerequisites	304
6.4.2. Internet access for OpenShift Container Platform	304
6.4.3. Generating a key pair for cluster node SSH access	305
6.4.4. Obtaining the installation program	307
6.4.5. Deploying the cluster	307
6.4.6. Installing the OpenShift CLI by downloading the binary	310
Installing the OpenShift CLI on Linux	310
Installing the OpenShift CLI on Windows	311
Installing the OpenShift CLI on macOS	311
6.4.7. Logging in to the cluster by using the CLI	312
6.4.8. Logging in to the cluster by using the web console	312
6.4.9. Telemetry access for OpenShift Container Platform	313
6.4.10. Next steps	313
6.5. INSTALLING A CLUSTER ON AWS WITH CUSTOMIZATIONS	314
6.5.1. Prerequisites	314
6.5.2. Internet access for OpenShift Container Platform	314
6.5.3. Generating a key pair for cluster node SSH access	315
6.5.4. Obtaining an AWS Marketplace image	317

6.5.5. Obtaining the installation program	317
6.5.6. Creating the installation configuration file	318
6.5.6.1. Installation configuration parameters	319
6.5.6.1.1. Required configuration parameters	320
6.5.6.1.2. Network configuration parameters	321
6.5.6.1.3. Optional configuration parameters	323
6.5.6.1.4. Optional AWS configuration parameters	329
6.5.6.2. Minimum resource requirements for cluster installation	333
6.5.6.3. Tested instance types for AWS	334
6.5.6.4. Tested instance types for AWS on 64-bit ARM infrastructures	335
6.5.6.5. Sample customized install-config.yaml file for AWS	335
6.5.6.6. Configuring the cluster-wide proxy during installation	338
6.5.7. Deploying the cluster	340
6.5.8. Installing the OpenShift CLI by downloading the binary	341
Installing the OpenShift CLI on Linux	341
Installing the OpenShift CLI on Windows	342
Installing the OpenShift CLI on macOS	342
6.5.9. Logging in to the cluster by using the CLI	343
6.5.10. Logging in to the cluster by using the web console	343
6.5.11. Telemetry access for OpenShift Container Platform	344
6.5.12. Next steps	345
6.6. INSTALLING A CLUSTER ON AWS WITH NETWORK CUSTOMIZATIONS	345
6.6.1. Prerequisites	345
6.6.2. Internet access for OpenShift Container Platform	346
6.6.3. Generating a key pair for cluster node SSH access	346
6.6.4. Obtaining the installation program	348
6.6.5. Network configuration phases	349
6.6.6. Creating the installation configuration file	349
6.6.6.1. Installation configuration parameters	351
6.6.6.1.1. Required configuration parameters	351
6.6.6.1.2. Network configuration parameters	352
6.6.6.1.3. Optional configuration parameters	354
6.6.6.1.4. Optional AWS configuration parameters	360
6.6.6.2. Minimum resource requirements for cluster installation	364
6.6.6.3. Tested instance types for AWS	365
6.6.6.4. Tested instance types for AWS on 64-bit ARM infrastructures	366
6.6.6.5. Sample customized install-config.yaml file for AWS	366
6.6.6.6. Configuring the cluster-wide proxy during installation	369
6.6.7. Cluster Network Operator configuration	371
6.6.7.1. Cluster Network Operator configuration object	371
defaultNetwork object configuration	372
Configuration for the OpenShift SDN network plugin	372
Configuration for the OVN-Kubernetes network plugin	373
kubeProxyConfig object configuration	377
6.6.8. Specifying advanced network configuration	378
6.6.9. Configuring an Ingress Controller Network Load Balancer on a new AWS cluster	379
6.6.10. Configuring hybrid networking with OVN-Kubernetes	380
6.6.11. Deploying the cluster	382
6.6.12. Installing the OpenShift CLI by downloading the binary	384
Installing the OpenShift CLI on Linux	384
Installing the OpenShift CLI on Windows	385
Installing the OpenShift CLI on macOS	385
6.6.13. Logging in to the cluster by using the CLI	386

6.6.14. Logging in to the cluster by using the web console	386
6.6.15. Telemetry access for OpenShift Container Platform	387
6.6.16. Next steps	387
6.7. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK	388
6.7.1. Prerequisites	388
6.7.2. About installations in restricted networks	389
6.7.2.1. Additional limits	389
6.7.3. About using a custom VPC	389
6.7.3.1. Requirements for using your VPC	389
Option 1: Create VPC endpoints	390
Option 2: Create a proxy without VPC endpoints	391
Option 3: Create a proxy with VPC endpoints	391
6.7.3.2. VPC validation	392
6.7.3.3. Division of permissions	393
6.7.3.4. Isolation between clusters	393
6.7.4. Internet access for OpenShift Container Platform	393
6.7.5. Generating a key pair for cluster node SSH access	394
6.7.6. Creating the installation configuration file	395
6.7.6.1. Installation configuration parameters	398
6.7.6.1.1. Required configuration parameters	398
6.7.6.1.2. Network configuration parameters	399
6.7.6.1.3. Optional configuration parameters	401
6.7.6.1.4. Optional AWS configuration parameters	407
6.7.6.2. Minimum resource requirements for cluster installation	411
6.7.6.3. Sample customized install-config.yaml file for AWS	412
6.7.6.4. Configuring the cluster-wide proxy during installation	415
6.7.7. Deploying the cluster	417
6.7.8. Installing the OpenShift CLI by downloading the binary	419
Installing the OpenShift CLI on Linux	419
Installing the OpenShift CLI on Windows	420
Installing the OpenShift CLI on macOS	420
6.7.9. Logging in to the cluster by using the CLI	421
6.7.10. Disabling the default OperatorHub catalog sources	421
6.7.11. Telemetry access for OpenShift Container Platform	422
6.7.12. Next steps	422
6.8. INSTALLING A CLUSTER ON AWS INTO AN EXISTING VPC	422
6.8.1. Prerequisites	422
6.8.2. About using a custom VPC	423
6.8.2.1. Requirements for using your VPC	423
Option 1: Create VPC endpoints	424
Option 2: Create a proxy without VPC endpoints	425
Option 3: Create a proxy with VPC endpoints	425
6.8.2.2. VPC validation	426
6.8.2.3. Division of permissions	427
6.8.2.4. Isolation between clusters	427
6.8.3. Internet access for OpenShift Container Platform	427
6.8.4. Generating a key pair for cluster node SSH access	428
6.8.5. Obtaining the installation program	430
6.8.6. Creating the installation configuration file	430
6.8.6.1. Installation configuration parameters	432
6.8.6.1.1. Required configuration parameters	432
6.8.6.1.2. Network configuration parameters	433
6.8.6.1.3. Optional configuration parameters	435

6.8.6.1.4. Optional AWS configuration parameters	441
6.8.6.2. Minimum resource requirements for cluster installation	445
6.8.6.3. Tested instance types for AWS	446
6.8.6.4. Tested instance types for AWS on 64-bit ARM infrastructures	447
6.8.6.5. Sample customized install-config.yaml file for AWS	447
6.8.6.6. Configuring the cluster-wide proxy during installation	450
6.8.7. Deploying the cluster	452
6.8.8. Installing the OpenShift CLI by downloading the binary	454
Installing the OpenShift CLI on Linux	454
Installing the OpenShift CLI on Windows	455
Installing the OpenShift CLI on macOS	455
6.8.9. Logging in to the cluster by using the CLI	456
6.8.10. Logging in to the cluster by using the web console	456
6.8.11. Telemetry access for OpenShift Container Platform	457
6.8.12. Next steps	457
6.9. INSTALLING A PRIVATE CLUSTER ON AWS	458
6.9.1. Prerequisites	458
6.9.2. Private clusters	458
6.9.2.1. Private clusters in AWS	459
6.9.2.1.1. Limitations	459
6.9.3. About using a custom VPC	459
6.9.3.1. Requirements for using your VPC	460
Option 1: Create VPC endpoints	461
Option 2: Create a proxy without VPC endpoints	461
Option 3: Create a proxy with VPC endpoints	461
6.9.3.2. VPC validation	463
6.9.3.3. Division of permissions	463
6.9.3.4. Isolation between clusters	463
6.9.4. Internet access for OpenShift Container Platform	463
6.9.5. Generating a key pair for cluster node SSH access	464
6.9.6. Obtaining the installation program	466
6.9.7. Manually creating the installation configuration file	466
6.9.7.1. Installation configuration parameters	467
6.9.7.1.1. Required configuration parameters	468
6.9.7.1.2. Network configuration parameters	469
6.9.7.1.3. Optional configuration parameters	471
6.9.7.1.4. Optional AWS configuration parameters	477
6.9.7.2. Minimum resource requirements for cluster installation	481
6.9.7.3. Tested instance types for AWS	482
6.9.7.4. Tested instance types for AWS on 64-bit ARM infrastructures	482
6.9.7.5. Sample customized install-config.yaml file for AWS	483
6.9.7.6. Configuring the cluster-wide proxy during installation	486
6.9.8. Deploying the cluster	487
6.9.9. Installing the OpenShift CLI by downloading the binary	489
Installing the OpenShift CLI on Linux	489
Installing the OpenShift CLI on Windows	490
Installing the OpenShift CLI on macOS	490
6.9.10. Logging in to the cluster by using the CLI	491
6.9.11. Logging in to the cluster by using the web console	491
6.9.12. Telemetry access for OpenShift Container Platform	492
6.9.13. Next steps	493
6.10. INSTALLING A CLUSTER ON AWS INTO A GOVERNMENT REGION	493
6.10.1. Prerequisites	493

6.10.2. AWS government regions	493
6.10.3. Installation requirements	494
6.10.4. Private clusters	494
6.10.4.1. Private clusters in AWS	494
6.10.4.1.1. Limitations	495
6.10.5. About using a custom VPC	495
6.10.5.1. Requirements for using your VPC	495
Option 1: Create VPC endpoints	496
Option 2: Create a proxy without VPC endpoints	496
Option 3: Create a proxy with VPC endpoints	497
6.10.5.2. VPC validation	498
6.10.5.3. Division of permissions	499
6.10.5.4. Isolation between clusters	499
6.10.6. Internet access for OpenShift Container Platform	499
6.10.7. Generating a key pair for cluster node SSH access	500
6.10.8. Obtaining an AWS Marketplace image	501
6.10.9. Obtaining the installation program	502
6.10.10. Manually creating the installation configuration file	503
6.10.10.1. Installation configuration parameters	504
6.10.10.1.1. Required configuration parameters	504
6.10.10.1.2. Network configuration parameters	506
6.10.10.1.3. Optional configuration parameters	508
6.10.10.1.4. Optional AWS configuration parameters	513
6.10.10.2. Minimum resource requirements for cluster installation	517
6.10.10.3. Tested instance types for AWS	518
6.10.10.4. Tested instance types for AWS on 64-bit ARM infrastructures	519
6.10.10.5. Sample customized install-config.yaml file for AWS	519
6.10.10.6. Configuring the cluster-wide proxy during installation	522
6.10.11. Deploying the cluster	524
6.10.12. Installing the OpenShift CLI by downloading the binary	526
Installing the OpenShift CLI on Linux	526
Installing the OpenShift CLI on Windows	527
Installing the OpenShift CLI on macOS	527
6.10.13. Logging in to the cluster by using the CLI	528
6.10.14. Logging in to the cluster by using the web console	528
6.10.15. Telemetry access for OpenShift Container Platform	529
6.10.16. Next steps	529
6.11. INSTALLING A CLUSTER ON AWS INTO A SECRET OR TOP SECRET REGION	530
6.11.1. Prerequisites	530
6.11.2. AWS secret regions	530
6.11.3. Installation requirements	531
6.11.4. Private clusters	531
6.11.4.1. Private clusters in AWS	532
6.11.4.1.1. Limitations	532
6.11.5. About using a custom VPC	532
6.11.5.1. Requirements for using your VPC	533
Option 1: Create VPC endpoints	534
Option 2: Create a proxy without VPC endpoints	534
Option 3: Create a proxy with VPC endpoints	534
6.11.5.2. VPC validation	536
6.11.5.3. Division of permissions	536
6.11.5.4. Isolation between clusters	536
6.11.6. Internet access for OpenShift Container Platform	537

6.11.7. Uploading a custom RHCOS AMI in AWS	537
6.11.8. Generating a key pair for cluster node SSH access	539
6.11.9. Obtaining the installation program	541
6.11.10. Manually creating the installation configuration file	542
6.11.10.1. Installation configuration parameters	543
6.11.10.1.1. Required configuration parameters	543
6.11.10.1.2. Network configuration parameters	545
6.11.10.1.3. Optional configuration parameters	547
6.11.10.1.4. Optional AWS configuration parameters	553
6.11.10.2. Tested instance types for AWS	557
6.11.10.3. Sample customized install-config.yaml file for AWS	557
6.11.10.4. Configuring the cluster-wide proxy during installation	560
6.11.11. Deploying the cluster	562
6.11.12. Installing the OpenShift CLI by downloading the binary	564
Installing the OpenShift CLI on Linux	564
Installing the OpenShift CLI on Windows	565
Installing the OpenShift CLI on macOS	565
6.11.13. Logging in to the cluster by using the CLI	566
6.11.14. Logging in to the cluster by using the web console	566
6.11.15. Telemetry access for OpenShift Container Platform	567
6.11.16. Next steps	567
6.12. INSTALLING A CLUSTER ON AWS CHINA	568
6.12.1. Prerequisites	568
6.12.2. Installation requirements	568
6.12.3. Internet access for OpenShift Container Platform	568
6.12.4. Private clusters	569
6.12.4.1. Private clusters in AWS	570
6.12.4.1.1. Limitations	570
6.12.5. About using a custom VPC	570
6.12.5.1. Requirements for using your VPC	570
Option 1: Create VPC endpoints	571
Option 2: Create a proxy without VPC endpoints	572
Option 3: Create a proxy with VPC endpoints	572
6.12.5.2. VPC validation	573
6.12.5.3. Division of permissions	574
6.12.5.4. Isolation between clusters	574
6.12.6. Generating a key pair for cluster node SSH access	574
6.12.7. Uploading a custom RHCOS AMI in AWS	576
6.12.8. Obtaining the installation program	578
6.12.9. Manually creating the installation configuration file	579
6.12.9.1. Installation configuration parameters	580
6.12.9.1.1. Required configuration parameters	580
6.12.9.1.2. Network configuration parameters	582
6.12.9.1.3. Optional configuration parameters	584
6.12.9.2. Sample customized install-config.yaml file for AWS	589
6.12.9.3. Minimum resource requirements for cluster installation	592
6.12.9.4. Tested instance types for AWS	593
6.12.9.5. Tested instance types for AWS on 64-bit ARM infrastructures	594
6.12.9.6. Configuring the cluster-wide proxy during installation	594
6.12.10. Deploying the cluster	596
6.12.11. Installing the OpenShift CLI by downloading the binary	598
Installing the OpenShift CLI on Linux	598
Installing the OpenShift CLI on Windows	599

Installing the OpenShift CLI on macOS	599
6.12.12. Logging in to the cluster by using the CLI	600
6.12.13. Logging in to the cluster by using the web console	600
6.12.14. Telemetry access for OpenShift Container Platform	601
6.12.15. Next steps	601
6.13. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN AWS BY USING CLOUDFORMATION TEMPLATES	602
6.13.1. Prerequisites	602
6.13.2. Internet access for OpenShift Container Platform	603
6.13.3. Requirements for a cluster with user-provisioned infrastructure	603
6.13.3.1. Required machines for cluster installation	603
6.13.3.2. Minimum resource requirements for cluster installation	604
6.13.3.3. Tested instance types for AWS	604
6.13.3.4. Tested instance types for AWS on 64-bit ARM infrastructures	605
6.13.3.5. Certificate signing requests management	606
6.13.4. Required AWS infrastructure components	606
6.13.4.1. Other infrastructure components	606
Option 1: Create VPC endpoints	607
Option 2: Create a proxy without VPC endpoints	607
Option 3: Create a proxy with VPC endpoints	607
6.13.4.2. Cluster machines	614
6.13.4.3. Required AWS permissions for the IAM user	615
6.13.5. Obtaining an AWS Marketplace image	622
6.13.6. Obtaining the installation program	623
6.13.7. Generating a key pair for cluster node SSH access	624
6.13.8. Creating the installation files for AWS	625
6.13.8.1. Optional: Creating a separate /var partition	626
6.13.8.2. Creating the installation configuration file	628
6.13.8.3. Configuring the cluster-wide proxy during installation	630
6.13.8.4. Creating the Kubernetes manifest and Ignition config files	631
6.13.9. Extracting the infrastructure name	634
6.13.10. Creating a VPC in AWS	635
6.13.10.1. CloudFormation template for the VPC	636
6.13.11. Creating networking and load balancing components in AWS	642
6.13.11.1. CloudFormation template for the network and load balancers	646
6.13.12. Creating security group and roles in AWS	654
6.13.12.1. CloudFormation template for security objects	656
6.13.13. Accessing RHCOS AMIs with stream metadata	667
6.13.14. RHCOS AMIs for the AWS infrastructure	668
6.13.14.1. AWS regions without a published RHCOS AMI	670
6.13.14.2. Uploading a custom RHCOS AMI in AWS	671
6.13.15. Creating the bootstrap node in AWS	673
6.13.15.1. CloudFormation template for the bootstrap machine	678
6.13.16. Creating the control plane machines in AWS	682
6.13.16.1. CloudFormation template for control plane machines	686
6.13.17. Creating the worker nodes in AWS	692
6.13.17.1. CloudFormation template for worker machines	695
6.13.18. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure	698
6.13.19. Installing the OpenShift CLI by downloading the binary	699
Installing the OpenShift CLI on Linux	699
Installing the OpenShift CLI on Windows	700
Installing the OpenShift CLI on macOS	700
6.13.20. Logging in to the cluster by using the CLI	701

6.13.21. Approving the certificate signing requests for your machines	701
6.13.22. Initial Operator configuration	704
6.13.22.1. Image registry storage configuration	705
6.13.22.1.1. Configuring registry storage for AWS with user-provisioned infrastructure	705
6.13.22.1.2. Configuring storage for the image registry in non-production clusters	706
6.13.23. Deleting the bootstrap resources	707
6.13.24. Creating the Ingress DNS Records	707
6.13.25. Completing an AWS installation on user-provisioned infrastructure	710
6.13.26. Logging in to the cluster by using the web console	711
6.13.27. Telemetry access for OpenShift Container Platform	712
6.13.28. Additional resources	712
6.13.29. Next steps	712
6.14. INSTALLING A CLUSTER USING AWS LOCAL ZONES	712
6.14.1. Prerequisites	713
6.14.2. Cluster limitations in AWS Local Zones	713
6.14.3. Internet access for OpenShift Container Platform	714
6.14.4. Opting into AWS Local Zones	714
6.14.5. Obtaining an AWS Marketplace image	715
6.14.6. Creating a VPC that uses AWS Local Zones	716
6.14.6.1. CloudFormation template for the VPC that uses AWS Local Zones	718
6.14.7. Creating a subnet in AWS Local Zones	725
6.14.7.1. CloudFormation template for the subnet that uses AWS Local Zones	727
6.14.8. Obtaining the installation program	729
6.14.9. Generating a key pair for cluster node SSH access	730
6.14.10. Creating the installation files for AWS	731
6.14.10.1. Minimum resource requirements for cluster installation	731
6.14.10.2. Tested instance types for AWS	732
6.14.10.3. Creating the installation configuration file	733
6.14.10.4. Creating the Kubernetes manifest files	735
6.14.11. Deploying the cluster	738
6.14.12. Installing the OpenShift CLI by downloading the binary	740
Installing the OpenShift CLI on Linux	740
Installing the OpenShift CLI on Windows	741
Installing the OpenShift CLI on macOS	741
6.14.13. Logging in to the cluster by using the CLI	742
6.14.14. Logging in to the cluster by using the web console	742
6.14.15. Telemetry access for OpenShift Container Platform	743
6.14.16. Next steps	743
6.15. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE	744
6.15.1. Prerequisites	744
6.15.2. About installations in restricted networks	745
6.15.2.1. Additional limits	745
6.15.3. Internet access for OpenShift Container Platform	746
6.15.4. Requirements for a cluster with user-provisioned infrastructure	746
6.15.4.1. Required machines for cluster installation	746
6.15.4.2. Minimum resource requirements for cluster installation	747
6.15.4.3. Tested instance types for AWS	748
6.15.4.4. Tested instance types for AWS on 64-bit ARM infrastructures	748
6.15.4.5. Certificate signing requests management	749
6.15.5. Required AWS infrastructure components	749
6.15.5.1. Other infrastructure components	749
Option 1: Create VPC endpoints	750

Option 2: Create a proxy without VPC endpoints	750
Option 3: Create a proxy with VPC endpoints	750
6.15.5.2. Cluster machines	758
6.15.5.3. Required AWS permissions for the IAM user	758
6.15.6. Generating a key pair for cluster node SSH access	766
6.15.7. Creating the installation files for AWS	768
6.15.7.1. Optional: Creating a separate /var partition	768
6.15.7.2. Creating the installation configuration file	770
6.15.7.3. Configuring the cluster-wide proxy during installation	772
6.15.7.4. Creating the Kubernetes manifest and Ignition config files	774
6.15.8. Extracting the infrastructure name	777
6.15.9. Creating a VPC in AWS	777
6.15.9.1. CloudFormation template for the VPC	779
6.15.10. Creating networking and load balancing components in AWS	785
6.15.10.1. CloudFormation template for the network and load balancers	788
6.15.11. Creating security group and roles in AWS	796
6.15.11.1. CloudFormation template for security objects	799
6.15.12. Accessing RHCOS AMIs with stream metadata	810
6.15.13. RHCOS AMIs for the AWS infrastructure	811
6.15.14. Creating the bootstrap node in AWS	813
6.15.14.1. CloudFormation template for the bootstrap machine	818
6.15.15. Creating the control plane machines in AWS	822
6.15.15.1. CloudFormation template for control plane machines	827
6.15.16. Creating the worker nodes in AWS	832
6.15.16.1. CloudFormation template for worker machines	836
6.15.17. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure	838
6.15.18. Logging in to the cluster by using the CLI	839
6.15.19. Approving the certificate signing requests for your machines	839
6.15.20. Initial Operator configuration	842
6.15.20.1. Disabling the default OperatorHub catalog sources	843
6.15.20.2. Image registry storage configuration	844
6.15.20.2.1. Configuring registry storage for AWS with user-provisioned infrastructure	844
6.15.20.2.2. Configuring storage for the image registry in non-production clusters	845
6.15.21. Deleting the bootstrap resources	845
6.15.22. Creating the Ingress DNS Records	846
6.15.23. Completing an AWS installation on user-provisioned infrastructure	848
6.15.24. Logging in to the cluster by using the web console	849
6.15.25. Telemetry access for OpenShift Container Platform	850
6.15.26. Additional resources	851
6.15.27. Next steps	851
6.16. INSTALLING A CLUSTER ON AWS WITH REMOTE WORKERS ON AWS OUTPOSTS	851
6.16.1. Prerequisites	852
6.16.2. About using a custom VPC	852
6.16.2.1. Requirements for using your VPC	852
Option 1: Create VPC endpoints	854
Option 2: Create a proxy without VPC endpoints	855
Option 3: Create a proxy with VPC endpoints	855
6.16.2.2. VPC validation	856
6.16.2.3. Division of permissions	857
6.16.2.4. Isolation between clusters	857
6.16.3. Internet access for OpenShift Container Platform	857
6.16.4. Generating a key pair for cluster node SSH access	858
6.16.5. Obtaining the installation program	859

6.16.6. Minimum resource requirements for cluster installation	860
6.16.7. Identifying your AWS Outposts instance types	861
6.16.8. Creating the installation configuration file	862
6.16.8.1. Installation configuration parameters	863
6.16.8.1.1. Required configuration parameters	864
6.16.8.1.2. Network configuration parameters	865
6.16.8.1.3. Optional configuration parameters	867
6.16.8.1.4. Optional AWS configuration parameters	873
6.16.8.2. Sample customized install-config.yaml file for AWS	877
6.16.9. Generating manifest files	879
6.16.9.1. Modifying manifest files	880
6.16.10. Deploying the cluster	881
6.16.11. Installing the OpenShift CLI by downloading the binary	883
Installing the OpenShift CLI on Linux	883
Installing the OpenShift CLI on Windows	884
Installing the OpenShift CLI on macOS	884
6.16.12. Logging in to the cluster by using the CLI	885
6.16.13. Logging in to the cluster by using the web console	885
6.16.14. Telemetry access for OpenShift Container Platform	886
6.16.15. Cluster Limitations	886
6.16.16. Next steps	887
6.17. UNINSTALLING A CLUSTER ON AWS	887
6.17.1. Removing a cluster that uses installer-provisioned infrastructure	887
6.17.2. Deleting AWS resources with the Cloud Credential Operator utility	888
6.17.3. Deleting a cluster with a configured AWS Local Zone infrastructure	889
CHAPTER 7. INSTALLING ON AZURE	891
7.1. PREPARING TO INSTALL ON AZURE	891
7.1.1. Prerequisites	891
7.1.2. Requirements for installing OpenShift Container Platform on Azure	891
7.1.3. Choosing a method to install OpenShift Container Platform on Azure	891
7.1.3.1. Installing a cluster on installer-provisioned infrastructure	891
7.1.3.2. Installing a cluster on user-provisioned infrastructure	892
7.1.4. Next steps	892
7.2. CONFIGURING AN AZURE ACCOUNT	892
7.2.1. Azure account limits	892
7.2.2. Configuring a public DNS zone in Azure	895
7.2.3. Increasing Azure account limits	896
7.2.4. Required Azure roles	896
7.2.5. Required Azure permissions for installer-provisioned infrastructure	896
7.2.6. Creating a service principal	903
7.2.7. Supported Azure Marketplace regions	906
7.2.8. Supported Azure regions	906
Supported Azure public regions	906
Supported Azure Government regions	908
7.2.9. Next steps	908
7.3. MANUALLY CREATING IAM FOR AZURE	908
7.3.1. Alternatives to storing administrator-level secrets in the kube-system project	908
7.3.2. Manually create IAM	909
7.3.3. Next steps	911
7.4. ENABLING USER-MANAGED ENCRYPTION FOR AZURE	912
7.4.1. Preparing an Azure Disk Encryption Set	912
7.4.2. Next steps	914

7.5. INSTALLING A CLUSTER QUICKLY ON AZURE	914
7.5.1. Prerequisites	914
7.5.2. Internet access for OpenShift Container Platform	914
7.5.3. Generating a key pair for cluster node SSH access	915
7.5.4. Obtaining the installation program	917
7.5.5. Deploying the cluster	917
7.5.6. Installing the OpenShift CLI by downloading the binary	920
Installing the OpenShift CLI on Linux	920
Installing the OpenShift CLI on Windows	921
Installing the OpenShift CLI on macOS	921
7.5.7. Logging in to the cluster by using the CLI	922
7.5.8. Telemetry access for OpenShift Container Platform	923
7.5.9. Next steps	923
7.6. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS	923
7.6.1. Prerequisites	923
7.6.2. Internet access for OpenShift Container Platform	923
7.6.3. Generating a key pair for cluster node SSH access	924
7.6.4. Selecting an Azure Marketplace image	926
7.6.5. Obtaining the installation program	928
7.6.6. Creating the installation configuration file	929
7.6.6.1. Installation configuration parameters	930
7.6.6.1.1. Required configuration parameters	930
7.6.6.1.2. Network configuration parameters	932
7.6.6.1.3. Optional configuration parameters	934
7.6.6.1.4. Additional Azure configuration parameters	939
7.6.6.2. Minimum resource requirements for cluster installation	945
7.6.6.3. Tested instance types for Azure	946
7.6.6.4. Tested instance types for Azure on 64-bit ARM infrastructures	947
7.6.6.5. Sample customized install-config.yaml file for Azure	947
7.6.6.6. Configuring the cluster-wide proxy during installation	949
7.6.7. Deploying the cluster	951
7.6.8. Finalizing user-managed encryption after installation	953
7.6.9. Installing the OpenShift CLI by downloading the binary	955
Installing the OpenShift CLI on Linux	955
Installing the OpenShift CLI on Windows	956
Installing the OpenShift CLI on macOS	956
7.6.10. Logging in to the cluster by using the CLI	957
7.6.11. Telemetry access for OpenShift Container Platform	957
7.6.12. Next steps	958
7.7. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS	958
7.7.1. Prerequisites	958
7.7.2. Internet access for OpenShift Container Platform	958
7.7.3. Generating a key pair for cluster node SSH access	959
7.7.4. Obtaining the installation program	960
7.7.5. Creating the installation configuration file	961
7.7.5.1. Installation configuration parameters	963
7.7.5.1.1. Required configuration parameters	963
7.7.5.1.2. Network configuration parameters	965
7.7.5.1.3. Optional configuration parameters	967
7.7.5.1.4. Additional Azure configuration parameters	972
7.7.5.2. Minimum resource requirements for cluster installation	978
7.7.5.3. Tested instance types for Azure	979
7.7.5.4. Tested instance types for Azure on 64-bit ARM infrastructures	980

7.7.5.5. Sample customized install-config.yaml file for Azure	980
7.7.5.6. Configuring the cluster-wide proxy during installation	982
7.7.6. Network configuration phases	984
7.7.7. Specifying advanced network configuration	985
7.7.8. Cluster Network Operator configuration	986
7.7.8.1. Cluster Network Operator configuration object	986
defaultNetwork object configuration	987
Configuration for the OpenShift SDN network plugin	988
Configuration for the OVN-Kubernetes network plugin	989
kubeProxyConfig object configuration	993
7.7.9. Configuring hybrid networking with OVN-Kubernetes	994
7.7.10. Deploying the cluster	996
7.7.11. Finalizing user-managed encryption after installation	997
7.7.12. Installing the OpenShift CLI by downloading the binary	999
Installing the OpenShift CLI on Linux	999
Installing the OpenShift CLI on Windows	1000
Installing the OpenShift CLI on macOS	1001
7.7.13. Logging in to the cluster by using the CLI	1001
7.7.14. Telemetry access for OpenShift Container Platform	1002
7.7.15. Next steps	1002
7.8. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET	1002
7.8.1. Prerequisites	1002
7.8.2. About reusing a VNet for your OpenShift Container Platform cluster	1003
7.8.2.1. Requirements for using your VNet	1003
7.8.2.1.1. Network security group requirements	1004
7.8.2.2. Division of permissions	1005
7.8.2.3. Isolation between clusters	1005
7.8.3. Internet access for OpenShift Container Platform	1005
7.8.4. Generating a key pair for cluster node SSH access	1006
7.8.5. Obtaining the installation program	1008
7.8.6. Creating the installation configuration file	1008
7.8.6.1. Installation configuration parameters	1010
7.8.6.1.1. Required configuration parameters	1010
7.8.6.1.2. Network configuration parameters	1012
7.8.6.1.3. Optional configuration parameters	1014
7.8.6.1.4. Additional Azure configuration parameters	1019
7.8.6.2. Minimum resource requirements for cluster installation	1025
7.8.6.3. Tested instance types for Azure	1026
7.8.6.4. Tested instance types for Azure on 64-bit ARM infrastructures	1027
7.8.6.5. Sample customized install-config.yaml file for Azure	1027
7.8.6.6. Configuring the cluster-wide proxy during installation	1029
7.8.7. Deploying the cluster	1031
7.8.8. Finalizing user-managed encryption after installation	1033
7.8.9. Installing the OpenShift CLI by downloading the binary	1035
Installing the OpenShift CLI on Linux	1035
Installing the OpenShift CLI on Windows	1036
Installing the OpenShift CLI on macOS	1036
7.8.10. Logging in to the cluster by using the CLI	1037
7.8.11. Telemetry access for OpenShift Container Platform	1037
7.8.12. Next steps	1038
7.9. INSTALLING A PRIVATE CLUSTER ON AZURE	1038
7.9.1. Prerequisites	1038
7.9.2. Private clusters	1038

7.9.2.1. Private clusters in Azure	1039
7.9.2.1.1. Limitations	1039
7.9.2.2. User-defined outbound routing	1039
Private cluster with network address translation	1040
Private cluster with Azure Firewall	1040
Private cluster with a proxy configuration	1040
Private cluster with no internet access	1040
7.9.3. About reusing a VNet for your OpenShift Container Platform cluster	1041
7.9.3.1. Requirements for using your VNet	1041
7.9.3.1.1. Network security group requirements	1042
7.9.3.2. Division of permissions	1043
7.9.3.3. Isolation between clusters	1043
7.9.4. Internet access for OpenShift Container Platform	1043
7.9.5. Generating a key pair for cluster node SSH access	1044
7.9.6. Obtaining the installation program	1045
7.9.7. Manually creating the installation configuration file	1046
7.9.7.1. Installation configuration parameters	1047
7.9.7.1.1. Required configuration parameters	1047
7.9.7.1.2. Network configuration parameters	1049
7.9.7.1.3. Optional configuration parameters	1051
7.9.7.1.4. Additional Azure configuration parameters	1056
7.9.7.2. Minimum resource requirements for cluster installation	1062
7.9.7.3. Tested instance types for Azure	1063
7.9.7.4. Tested instance types for Azure on 64-bit ARM infrastructures	1064
7.9.7.5. Sample customized install-config.yaml file for Azure	1064
7.9.7.6. Configuring the cluster-wide proxy during installation	1067
7.9.8. Deploying the cluster	1068
7.9.9. Finalizing user-managed encryption after installation	1070
7.9.10. Installing the OpenShift CLI by downloading the binary	1072
Installing the OpenShift CLI on Linux	1072
Installing the OpenShift CLI on Windows	1073
Installing the OpenShift CLI on macOS	1073
7.9.11. Logging in to the cluster by using the CLI	1074
7.9.12. Telemetry access for OpenShift Container Platform	1074
7.9.13. Next steps	1075
7.10. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION	1075
7.10.1. Prerequisites	1075
7.10.2. Azure government regions	1075
7.10.3. Private clusters	1076
7.10.3.1. Private clusters in Azure	1076
7.10.3.1.1. Limitations	1077
7.10.3.2. User-defined outbound routing	1077
Private cluster with network address translation	1077
Private cluster with Azure Firewall	1078
Private cluster with a proxy configuration	1078
Private cluster with no internet access	1078
7.10.4. About reusing a VNet for your OpenShift Container Platform cluster	1078
7.10.4.1. Requirements for using your VNet	1078
7.10.4.1.1. Network security group requirements	1079
7.10.4.2. Division of permissions	1080
7.10.4.3. Isolation between clusters	1081
7.10.5. Internet access for OpenShift Container Platform	1081
7.10.6. Generating a key pair for cluster node SSH access	1081

7.10.7. Obtaining the installation program	1083
7.10.8. Manually creating the installation configuration file	1084
7.10.8.1. Installation configuration parameters	1085
7.10.8.1.1. Required configuration parameters	1085
7.10.8.1.2. Network configuration parameters	1086
7.10.8.1.3. Optional configuration parameters	1088
7.10.8.1.4. Additional Azure configuration parameters	1094
7.10.8.2. Minimum resource requirements for cluster installation	1100
7.10.8.3. Tested instance types for Azure	1101
7.10.8.4. Sample customized install-config.yaml file for Azure	1102
7.10.8.5. Configuring the cluster-wide proxy during installation	1105
7.10.9. Deploying the cluster	1106
7.10.10. Finalizing user-managed encryption after installation	1108
7.10.11. Installing the OpenShift CLI by downloading the binary	1110
Installing the OpenShift CLI on Linux	1110
Installing the OpenShift CLI on Windows	1111
Installing the OpenShift CLI on macOS	1111
7.10.12. Logging in to the cluster by using the CLI	1112
7.10.13. Telemetry access for OpenShift Container Platform	1112
7.10.14. Next steps	1113
7.11. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES	1113
7.11.1. Prerequisites	1113
7.11.2. Internet access for OpenShift Container Platform	1114
7.11.3. Configuring your Azure project	1114
7.11.3.1. Azure account limits	1114
7.11.3.2. Configuring a public DNS zone in Azure	1117
7.11.3.3. Increasing Azure account limits	1117
7.11.3.4. Certificate signing requests management	1118
7.11.3.5. Required Azure roles	1118
7.11.3.6. Required Azure permissions for user-provisioned infrastructure	1119
7.11.3.7. Creating a service principal	1124
7.11.3.8. Supported Azure regions	1127
Supported Azure public regions	1127
Supported Azure Government regions	1128
7.11.4. Requirements for a cluster with user-provisioned infrastructure	1129
7.11.4.1. Required machines for cluster installation	1129
7.11.4.2. Minimum resource requirements for cluster installation	1129
7.11.4.3. Tested instance types for Azure	1130
7.11.5. Selecting an Azure Marketplace image	1131
7.11.6. Obtaining the installation program	1133
7.11.7. Generating a key pair for cluster node SSH access	1134
7.11.8. Creating the installation files for Azure	1136
7.11.8.1. Optional: Creating a separate /var partition	1136
7.11.8.2. Creating the installation configuration file	1138
7.11.8.3. Configuring the cluster-wide proxy during installation	1140
7.11.8.4. Exporting common variables for ARM templates	1142
7.11.8.5. Creating the Kubernetes manifest and Ignition config files	1143
7.11.9. Creating the Azure resource group	1146
7.11.10. Uploading the RHCOS cluster image and bootstrap Ignition config file	1146
7.11.11. Example for creating DNS zones	1148
7.11.12. Creating a VNet in Azure	1149
7.11.12.1. ARM template for the VNet	1149
7.11.13. Deploying the RHCOS cluster image for the Azure infrastructure	1151

7.11.13.1. ARM template for image storage	1152
7.11.14. Networking requirements for user-provisioned infrastructure	1155
7.11.14.1. Network connectivity requirements	1155
7.11.15. Creating networking and load balancing components in Azure	1156
7.11.15.1. ARM template for the network and load balancers	1157
7.11.16. Creating the bootstrap machine in Azure	1162
7.11.16.1. ARM template for the bootstrap machine	1163
7.11.17. Creating the control plane machines in Azure	1167
7.11.17.1. ARM template for control plane machines	1168
7.11.18. Wait for bootstrap completion and remove bootstrap resources in Azure	1172
7.11.19. Creating additional worker machines in Azure	1173
7.11.19.1. ARM template for worker machines	1174
7.11.20. Installing the OpenShift CLI by downloading the binary	1177
Installing the OpenShift CLI on Linux	1178
Installing the OpenShift CLI on Windows	1178
Installing the OpenShift CLI on macOS	1179
7.11.21. Logging in to the cluster by using the CLI	1179
7.11.22. Approving the certificate signing requests for your machines	1180
7.11.23. Adding the Ingress DNS records	1182
7.11.24. Completing an Azure installation on user-provisioned infrastructure	1184
7.11.25. Telemetry access for OpenShift Container Platform	1184
7.12. UNINSTALLING A CLUSTER ON AZURE	1185
7.12.1. Removing a cluster that uses installer-provisioned infrastructure	1185
CHAPTER 8. INSTALLING ON AZURE STACK HUB	1187
8.1. PREPARING TO INSTALL ON AZURE STACK HUB	1187
8.1.1. Prerequisites	1187
8.1.2. Requirements for installing OpenShift Container Platform on Azure Stack Hub	1187
8.1.3. Choosing a method to install OpenShift Container Platform on Azure Stack Hub	1187
8.1.3.1. Installing a cluster on installer-provisioned infrastructure	1187
8.1.3.2. Installing a cluster on user-provisioned infrastructure	1187
8.1.4. Next steps	1187
8.2. CONFIGURING AN AZURE STACK HUB ACCOUNT	1188
8.2.1. Azure Stack Hub account limits	1188
8.2.2. Configuring a DNS zone in Azure Stack Hub	1189
8.2.3. Required Azure Stack Hub roles	1190
8.2.4. Creating a service principal	1190
8.2.5. Next steps	1192
8.3. INSTALLING A CLUSTER ON AZURE STACK HUB WITH AN INSTALLER-PROVISIONED INFRASTRUCTURE	1193
8.3.1. Prerequisites	1193
8.3.2. Internet access for OpenShift Container Platform	1193
8.3.3. Generating a key pair for cluster node SSH access	1194
8.3.4. Uploading the RHCOS cluster image	1195
8.3.5. Obtaining the installation program	1196
8.3.6. Manually creating the installation configuration file	1197
8.3.6.1. Installation configuration parameters	1198
8.3.6.1.1. Required configuration parameters	1198
8.3.6.1.2. Network configuration parameters	1199
8.3.6.1.3. Optional configuration parameters	1201
8.3.6.1.4. Additional Azure Stack Hub configuration parameters	1206
8.3.6.2. Sample customized install-config.yaml file for Azure Stack Hub	1208
8.3.7. Manually manage cloud credentials	1210

8.3.8. Configuring the cluster to use an internal CA	1213
8.3.9. Deploying the cluster	1213
8.3.10. Installing the OpenShift CLI by downloading the binary	1215
Installing the OpenShift CLI on Linux	1215
Installing the OpenShift CLI on Windows	1216
Installing the OpenShift CLI on macOS	1216
8.3.11. Logging in to the cluster by using the CLI	1217
8.3.12. Logging in to the cluster by using the web console	1217
8.3.13. Telemetry access for OpenShift Container Platform	1218
8.3.14. Next steps	1218
8.4. INSTALLING A CLUSTER ON AZURE STACK HUB WITH NETWORK CUSTOMIZATIONS	1219
8.4.1. Prerequisites	1219
8.4.2. Internet access for OpenShift Container Platform	1219
8.4.3. Generating a key pair for cluster node SSH access	1220
8.4.4. Uploading the RHCOS cluster image	1221
8.4.5. Obtaining the installation program	1222
8.4.6. Manually creating the installation configuration file	1223
8.4.6.1. Installation configuration parameters	1224
8.4.6.1.1. Required configuration parameters	1224
8.4.6.1.2. Network configuration parameters	1225
8.4.6.1.3. Optional configuration parameters	1227
8.4.6.1.4. Additional Azure Stack Hub configuration parameters	1232
8.4.6.2. Sample customized install-config.yaml file for Azure Stack Hub	1234
8.4.7. Manually manage cloud credentials	1236
8.4.8. Configuring the cluster to use an internal CA	1239
8.4.9. Network configuration phases	1239
8.4.10. Specifying advanced network configuration	1240
8.4.11. Cluster Network Operator configuration	1241
8.4.11.1. Cluster Network Operator configuration object	1242
defaultNetwork object configuration	1242
Configuration for the OpenShift SDN network plugin	1243
Configuration for the OVN-Kubernetes network plugin	1244
kubeProxyConfig object configuration	1248
8.4.12. Configuring hybrid networking with OVN-Kubernetes	1249
8.4.13. Deploying the cluster	1251
8.4.14. Installing the OpenShift CLI by downloading the binary	1252
Installing the OpenShift CLI on Linux	1252
Installing the OpenShift CLI on Windows	1253
Installing the OpenShift CLI on macOS	1253
8.4.15. Logging in to the cluster by using the CLI	1254
8.4.16. Logging in to the cluster by using the web console	1255
8.4.17. Telemetry access for OpenShift Container Platform	1255
8.4.18. Next steps	1256
8.5. INSTALLING A CLUSTER ON AZURE STACK HUB USING ARM TEMPLATES	1256
8.5.1. Prerequisites	1256
8.5.2. Internet access for OpenShift Container Platform	1257
8.5.3. Configuring your Azure Stack Hub project	1257
8.5.3.1. Azure Stack Hub account limits	1257
8.5.3.2. Configuring a DNS zone in Azure Stack Hub	1259
8.5.3.3. Certificate signing requests management	1259
8.5.3.4. Required Azure Stack Hub roles	1259
8.5.3.5. Creating a service principal	1260
8.5.4. Obtaining the installation program	1262

8.5.5. Generating a key pair for cluster node SSH access	1263
8.5.6. Creating the installation files for Azure Stack Hub	1265
8.5.6.1. Manually creating the installation configuration file	1265
8.5.6.2. Sample customized install-config.yaml file for Azure Stack Hub	1267
8.5.6.3. Configuring the cluster-wide proxy during installation	1269
8.5.6.4. Exporting common variables for ARM templates	1270
8.5.6.5. Creating the Kubernetes manifest and Ignition config files	1271
8.5.6.6. Optional: Creating a separate /var partition	1276
8.5.7. Creating the Azure resource group	1278
8.5.8. Uploading the RHCOS cluster image and bootstrap Ignition config file	1279
8.5.9. Example for creating DNS zones	1280
8.5.10. Creating a VNet in Azure Stack Hub	1281
8.5.10.1. ARM template for the VNet	1281
8.5.11. Deploying the RHCOS cluster image for the Azure Stack Hub infrastructure	1282
8.5.11.1. ARM template for image storage	1282
8.5.12. Networking requirements for user-provisioned infrastructure	1283
8.5.12.1. Network connectivity requirements	1283
8.5.13. Creating networking and load balancing components in Azure Stack Hub	1284
8.5.13.1. ARM template for the network and load balancers	1285
8.5.14. Creating the bootstrap machine in Azure Stack Hub	1286
8.5.14.1. ARM template for the bootstrap machine	1287
8.5.15. Creating the control plane machines in Azure Stack Hub	1287
8.5.15.1. ARM template for control plane machines	1288
8.5.16. Wait for bootstrap completion and remove bootstrap resources in Azure Stack Hub	1288
8.5.17. Creating additional worker machines in Azure Stack Hub	1289
8.5.17.1. ARM template for worker machines	1290
8.5.18. Installing the OpenShift CLI by downloading the binary	1291
Installing the OpenShift CLI on Linux	1291
Installing the OpenShift CLI on Windows	1291
Installing the OpenShift CLI on macOS	1292
8.5.19. Logging in to the cluster by using the CLI	1292
8.5.20. Approving the certificate signing requests for your machines	1293
8.5.21. Adding the Ingress DNS records	1296
8.5.22. Completing an Azure Stack Hub installation on user-provisioned infrastructure	1297
8.6. UNINSTALLING A CLUSTER ON AZURE STACK HUB	1298
8.6.1. Removing a cluster that uses installer-provisioned infrastructure	1298
CHAPTER 9. INSTALLING ON GCP	1300
9.1. PREPARING TO INSTALL ON GCP	1300
9.1.1. Prerequisites	1300
9.1.2. Requirements for installing OpenShift Container Platform on GCP	1300
9.1.3. Choosing a method to install OpenShift Container Platform on GCP	1300
9.1.3.1. Installing a cluster on installer-provisioned infrastructure	1300
9.1.3.2. Installing a cluster on user-provisioned infrastructure	1301
9.1.4. Next steps	1301
9.2. CONFIGURING A GCP PROJECT	1301
9.2.1. Creating a GCP project	1301
9.2.2. Enabling API services in GCP	1302
9.2.3. Configuring DNS for GCP	1303
9.2.4. GCP account limits	1303
9.2.5. Creating a service account in GCP	1305
9.2.5.1. Required GCP roles	1306
9.2.5.2. Required GCP permissions for installer-provisioned infrastructure	1307

9.2.6. Supported GCP regions	1314
9.2.7. Next steps	1315
9.3. MANUALLY CREATING IAM FOR GCP	1315
9.3.1. Alternatives to storing administrator-level secrets in the kube-system project	1315
9.3.2. Manually create IAM	1316
9.3.3. Mint mode	1319
9.3.4. Mint mode with removal or rotation of the administrator-level credential	1320
9.3.5. Next steps	1320
9.4. INSTALLING A CLUSTER QUICKLY ON GCP	1320
9.4.1. Prerequisites	1320
9.4.2. Internet access for OpenShift Container Platform	1321
9.4.3. Generating a key pair for cluster node SSH access	1321
9.4.4. Obtaining the installation program	1323
9.4.5. Deploying the cluster	1324
9.4.6. Installing the OpenShift CLI by downloading the binary	1326
Installing the OpenShift CLI on Linux	1326
Installing the OpenShift CLI on Windows	1327
Installing the OpenShift CLI on macOS	1327
9.4.7. Logging in to the cluster by using the CLI	1328
9.4.8. Telemetry access for OpenShift Container Platform	1329
9.4.9. Next steps	1329
9.5. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS	1329
9.5.1. Prerequisites	1329
9.5.2. Internet access for OpenShift Container Platform	1329
9.5.3. Generating a key pair for cluster node SSH access	1330
9.5.4. Obtaining the installation program	1332
9.5.5. Creating the installation configuration file	1332
9.5.5.1. Installation configuration parameters	1334
9.5.5.1.1. Required configuration parameters	1334
9.5.5.1.2. Network configuration parameters	1335
9.5.5.1.3. Optional configuration parameters	1337
9.5.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters	1342
9.5.5.2. Minimum resource requirements for cluster installation	1352
9.5.5.3. Tested instance types for GCP	1353
9.5.5.4. Using custom machine types	1353
9.5.5.5. Sample customized install-config.yaml file for GCP	1354
9.5.5.6. Configuring the cluster-wide proxy during installation	1357
9.5.6. Using the GCP Marketplace offering	1359
9.5.7. Deploying the cluster	1360
9.5.8. Installing the OpenShift CLI by downloading the binary	1361
Installing the OpenShift CLI on Linux	1362
Installing the OpenShift CLI on Windows	1362
Installing the OpenShift CLI on macOS	1363
9.5.9. Logging in to the cluster by using the CLI	1363
9.5.10. Telemetry access for OpenShift Container Platform	1364
9.5.11. Next steps	1364
9.6. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS	1364
9.6.1. Prerequisites	1364
9.6.2. Internet access for OpenShift Container Platform	1365
9.6.3. Generating a key pair for cluster node SSH access	1365
9.6.4. Obtaining the installation program	1367
9.6.5. Creating the installation configuration file	1368
9.6.5.1. Installation configuration parameters	1369

9.6.5.1.1. Required configuration parameters	1369
9.6.5.1.2. Network configuration parameters	1371
9.6.5.1.3. Optional configuration parameters	1373
9.6.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters	1378
9.6.5.2. Minimum resource requirements for cluster installation	1388
9.6.5.3. Tested instance types for GCP	1389
9.6.5.4. Using custom machine types	1389
9.6.5.5. Sample customized install-config.yaml file for GCP	1390
9.6.5.6. Configuring the cluster-wide proxy during installation	1393
9.6.6. Network configuration phases	1395
9.6.7. Specifying advanced network configuration	1395
9.6.8. Cluster Network Operator configuration	1396
9.6.8.1. Cluster Network Operator configuration object	1397
defaultNetwork object configuration	1398
Configuration for the OpenShift SDN network plugin	1398
Configuration for the OVN-Kubernetes network plugin	1399
kubeProxyConfig object configuration	1403
9.6.9. Deploying the cluster	1404
9.6.10. Installing the OpenShift CLI by downloading the binary	1406
Installing the OpenShift CLI on Linux	1406
Installing the OpenShift CLI on Windows	1407
Installing the OpenShift CLI on macOS	1407
9.6.11. Logging in to the cluster by using the CLI	1408
9.6.12. Telemetry access for OpenShift Container Platform	1408
9.6.13. Next steps	1409
9.7. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK	1409
9.7.1. Prerequisites	1409
9.7.2. About installations in restricted networks	1410
9.7.2.1. Additional limits	1410
9.7.3. Internet access for OpenShift Container Platform	1410
9.7.4. Generating a key pair for cluster node SSH access	1410
9.7.5. Creating the installation configuration file	1412
9.7.5.1. Installation configuration parameters	1415
9.7.5.1.1. Required configuration parameters	1415
9.7.5.1.2. Network configuration parameters	1416
9.7.5.1.3. Optional configuration parameters	1418
9.7.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters	1423
9.7.5.2. Minimum resource requirements for cluster installation	1433
9.7.5.3. Tested instance types for GCP	1434
9.7.5.4. Using custom machine types	1434
9.7.5.5. Sample customized install-config.yaml file for GCP	1435
9.7.5.6. Create an Ingress Controller with global access on GCP	1438
9.7.5.7. Configuring the cluster-wide proxy during installation	1440
9.7.6. Deploying the cluster	1441
9.7.7. Installing the OpenShift CLI by downloading the binary	1443
Installing the OpenShift CLI on Linux	1443
Installing the OpenShift CLI on Windows	1444
Installing the OpenShift CLI on macOS	1444
9.7.8. Logging in to the cluster by using the CLI	1445
9.7.9. Disabling the default OperatorHub catalog sources	1445
9.7.10. Telemetry access for OpenShift Container Platform	1446
9.7.11. Next steps	1446
9.8. INSTALLING A CLUSTER ON GCP INTO AN EXISTING VPC	1446

9.8.1. Prerequisites	1446
9.8.2. About using a custom VPC	1447
9.8.2.1. Requirements for using your VPC	1447
9.8.2.2. VPC validation	1447
9.8.2.3. Division of permissions	1447
9.8.2.4. Isolation between clusters	1447
9.8.3. Internet access for OpenShift Container Platform	1448
9.8.4. Generating a key pair for cluster node SSH access	1448
9.8.5. Obtaining the installation program	1450
9.8.6. Creating the installation configuration file	1451
9.8.6.1. Installation configuration parameters	1452
9.8.6.1.1. Required configuration parameters	1452
9.8.6.1.2. Network configuration parameters	1454
9.8.6.1.3. Optional configuration parameters	1456
9.8.6.1.4. Additional Google Cloud Platform (GCP) configuration parameters	1461
9.8.6.2. Minimum resource requirements for cluster installation	1471
9.8.6.3. Tested instance types for GCP	1472
9.8.6.4. Using custom machine types	1472
9.8.6.5. Sample customized install-config.yaml file for GCP	1473
9.8.6.6. Create an Ingress Controller with global access on GCP	1476
9.8.6.7. Configuring the cluster-wide proxy during installation	1477
9.8.7. Deploying the cluster	1479
9.8.8. Installing the OpenShift CLI by downloading the binary	1481
Installing the OpenShift CLI on Linux	1481
Installing the OpenShift CLI on Windows	1482
Installing the OpenShift CLI on macOS	1482
9.8.9. Logging in to the cluster by using the CLI	1483
9.8.10. Telemetry access for OpenShift Container Platform	1483
9.8.11. Next steps	1484
9.9. INSTALLING A CLUSTER ON GCP INTO A SHARED VPC	1484
9.9.1. Prerequisites	1484
9.9.2. Internet access for OpenShift Container Platform	1485
9.9.3. Generating a key pair for cluster node SSH access	1485
9.9.4. Obtaining the installation program	1487
9.9.5. Creating the installation files for GCP	1488
9.9.5.1. Manually creating the installation configuration file	1488
9.9.5.2. Sample customized install-config.yaml file for shared VPC installation	1488
9.9.5.3. Installation configuration parameters	1490
9.9.5.3.1. Required configuration parameters	1491
9.9.5.3.2. Network configuration parameters	1492
9.9.5.3.3. Optional configuration parameters	1494
9.9.5.3.4. Additional Google Cloud Platform (GCP) configuration parameters	1499
9.9.5.4. Configuring the cluster-wide proxy during installation	1509
9.9.6. Deploying the cluster	1511
9.9.7. Installing the OpenShift CLI by downloading the binary	1513
Installing the OpenShift CLI on Linux	1513
Installing the OpenShift CLI on Windows	1514
Installing the OpenShift CLI on macOS	1514
9.9.8. Logging in to the cluster by using the CLI	1515
9.9.9. Optional: Adding Ingress DNS records for shared VPC installations	1515
9.9.10. Telemetry access for OpenShift Container Platform	1516
9.9.11. Next steps	1516
9.10. INSTALLING A PRIVATE CLUSTER ON GCP	1517

9.10.1. Prerequisites	1517
9.10.2. Private clusters	1517
9.10.2.1. Private clusters in GCP	1518
9.10.2.1.1. Limitations	1518
9.10.3. About using a custom VPC	1518
9.10.3.1. Requirements for using your VPC	1519
9.10.3.2. Division of permissions	1519
9.10.3.3. Isolation between clusters	1520
9.10.4. Internet access for OpenShift Container Platform	1520
9.10.5. Generating a key pair for cluster node SSH access	1520
9.10.6. Obtaining the installation program	1522
9.10.7. Manually creating the installation configuration file	1523
9.10.7.1. Installation configuration parameters	1524
9.10.7.1.1. Required configuration parameters	1524
9.10.7.1.2. Network configuration parameters	1526
9.10.7.1.3. Optional configuration parameters	1528
9.10.7.1.4. Additional Google Cloud Platform (GCP) configuration parameters	1533
9.10.7.2. Minimum resource requirements for cluster installation	1543
9.10.7.3. Tested instance types for GCP	1544
9.10.7.4. Using custom machine types	1544
9.10.7.5. Sample customized install-config.yaml file for GCP	1545
9.10.7.6. Create an Ingress Controller with global access on GCP	1548
9.10.7.7. Configuring the cluster-wide proxy during installation	1550
9.10.8. Deploying the cluster	1551
9.10.9. Installing the OpenShift CLI by downloading the binary	1553
Installing the OpenShift CLI on Linux	1553
Installing the OpenShift CLI on Windows	1554
Installing the OpenShift CLI on macOS	1554
9.10.10. Logging in to the cluster by using the CLI	1555
9.10.11. Telemetry access for OpenShift Container Platform	1555
9.10.12. Next steps	1556
9.11. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN GCP BY USING DEPLOYMENT MANAGER TEMPLATES	1556
9.11.1. Prerequisites	1556
9.11.2. Certificate signing requests management	1557
9.11.3. Internet access for OpenShift Container Platform	1557
9.11.4. Configuring your GCP project	1557
9.11.4.1. Creating a GCP project	1557
9.11.4.2. Enabling API services in GCP	1558
9.11.4.3. Configuring DNS for GCP	1558
9.11.4.4. GCP account limits	1559
9.11.4.5. Creating a service account in GCP	1561
9.11.4.6. Required GCP roles	1561
9.11.4.7. Required GCP permissions for user-provisioned infrastructure	1563
9.11.4.8. Supported GCP regions	1570
9.11.4.9. Installing and configuring CLI tools for GCP	1572
9.11.5. Requirements for a cluster with user-provisioned infrastructure	1572
9.11.5.1. Required machines for cluster installation	1572
9.11.5.2. Minimum resource requirements for cluster installation	1573
9.11.5.3. Tested instance types for GCP	1573
9.11.5.4. Using custom machine types	1574
9.11.6. Creating the installation files for GCP	1574
9.11.6.1. Optional: Creating a separate /var partition	1574

9.11.6.2. Creating the installation configuration file	1577
9.11.6.3. Configuring the cluster-wide proxy during installation	1578
9.11.6.4. Creating the Kubernetes manifest and Ignition config files	1580
9.11.7. Exporting common variables	1582
9.11.7.1. Extracting the infrastructure name	1582
9.11.7.2. Exporting common variables for Deployment Manager templates	1583
9.11.8. Creating a VPC in GCP	1584
9.11.8.1. Deployment Manager template for the VPC	1585
9.11.9. Networking requirements for user-provisioned infrastructure	1586
9.11.9.1. Setting the cluster node hostnames through DHCP	1586
9.11.9.2. Network connectivity requirements	1586
9.11.10. Creating load balancers in GCP	1587
9.11.10.1. Deployment Manager template for the external load balancer	1589
9.11.10.2. Deployment Manager template for the internal load balancer	1590
9.11.11. Creating a private DNS zone in GCP	1592
9.11.11.1. Deployment Manager template for the private DNS	1593
9.11.12. Creating firewall rules in GCP	1594
9.11.12.1. Deployment Manager template for firewall rules	1595
9.11.13. Creating IAM roles in GCP	1597
9.11.13.1. Deployment Manager template for IAM roles	1599
9.11.14. Creating the RHCOS cluster image for the GCP infrastructure	1599
9.11.15. Creating the bootstrap machine in GCP	1600
9.11.15.1. Deployment Manager template for the bootstrap machine	1602
9.11.16. Creating the control plane machines in GCP	1603
9.11.16.1. Deployment Manager template for control plane machines	1605
9.11.17. Wait for bootstrap completion and remove bootstrap resources in GCP	1608
9.11.18. Creating additional worker machines in GCP	1609
9.11.18.1. Deployment Manager template for worker machines	1611
9.11.19. Installing the OpenShift CLI by downloading the binary	1612
Installing the OpenShift CLI on Linux	1612
Installing the OpenShift CLI on Windows	1612
Installing the OpenShift CLI on macOS	1613
9.11.20. Logging in to the cluster by using the CLI	1613
9.11.21. Approving the certificate signing requests for your machines	1614
9.11.22. Optional: Adding the ingress DNS records	1617
9.11.23. Completing a GCP installation on user-provisioned infrastructure	1618
9.11.24. Telemetry access for OpenShift Container Platform	1621
9.11.25. Next steps	1621
9.12. INSTALLING A CLUSTER INTO A SHARED VPC ON GCP USING DEPLOYMENT MANAGER TEMPLATES	1621
9.12.1. Prerequisites	1622
9.12.2. Certificate signing requests management	1622
9.12.3. Internet access for OpenShift Container Platform	1622
9.12.4. Configuring the GCP project that hosts your cluster	1623
9.12.4.1. Creating a GCP project	1623
9.12.4.2. Enabling API services in GCP	1623
9.12.4.3. GCP account limits	1624
9.12.4.4. Creating a service account in GCP	1626
9.12.4.4.1. Required GCP roles	1626
9.12.4.5. Supported GCP regions	1627
9.12.4.6. Installing and configuring CLI tools for GCP	1629
9.12.5. Requirements for a cluster with user-provisioned infrastructure	1629
9.12.5.1. Required machines for cluster installation	1629

9.12.5.2. Minimum resource requirements for cluster installation	1630
9.12.5.3. Tested instance types for GCP	1631
9.12.5.4. Using custom machine types	1631
9.12.6. Configuring the GCP project that hosts your shared VPC network	1632
9.12.6.1. Configuring DNS for GCP	1632
9.12.6.2. Creating a VPC in GCP	1633
9.12.6.2.1. Deployment Manager template for the VPC	1635
9.12.7. Creating the installation files for GCP	1636
9.12.7.1. Manually creating the installation configuration file	1636
9.12.7.2. Sample customized install-config.yaml file for GCP	1637
9.12.7.3. Configuring the cluster-wide proxy during installation	1639
9.12.7.4. Creating the Kubernetes manifest and Ignition config files	1641
9.12.8. Exporting common variables	1644
9.12.8.1. Extracting the infrastructure name	1644
9.12.8.2. Exporting common variables for Deployment Manager templates	1644
9.12.9. Networking requirements for user-provisioned infrastructure	1645
9.12.9.1. Setting the cluster node hostnames through DHCP	1645
9.12.9.2. Network connectivity requirements	1646
9.12.10. Creating load balancers in GCP	1647
9.12.10.1. Deployment Manager template for the external load balancer	1649
9.12.10.2. Deployment Manager template for the internal load balancer	1650
9.12.11. Creating a private DNS zone in GCP	1651
9.12.11.1. Deployment Manager template for the private DNS	1653
9.12.12. Creating firewall rules in GCP	1653
9.12.12.1. Deployment Manager template for firewall rules	1654
9.12.13. Creating IAM roles in GCP	1657
9.12.13.1. Deployment Manager template for IAM roles	1659
9.12.14. Creating the RHCOS cluster image for the GCP infrastructure	1660
9.12.15. Creating the bootstrap machine in GCP	1660
9.12.15.1. Deployment Manager template for the bootstrap machine	1662
9.12.16. Creating the control plane machines in GCP	1664
9.12.16.1. Deployment Manager template for control plane machines	1666
9.12.17. Wait for bootstrap completion and remove bootstrap resources in GCP	1668
9.12.18. Creating additional worker machines in GCP	1669
9.12.18.1. Deployment Manager template for worker machines	1671
9.12.19. Installing the OpenShift CLI by downloading the binary	1672
Installing the OpenShift CLI on Linux	1672
Installing the OpenShift CLI on Windows	1673
Installing the OpenShift CLI on macOS	1673
9.12.20. Logging in to the cluster by using the CLI	1674
9.12.21. Approving the certificate signing requests for your machines	1674
9.12.22. Adding the ingress DNS records	1677
9.12.23. Adding ingress firewall rules	1679
9.12.23.1. Creating cluster-wide firewall rules for a shared VPC in GCP	1679
9.12.24. Completing a GCP installation on user-provisioned infrastructure	1680
9.12.25. Telemetry access for OpenShift Container Platform	1683
9.12.26. Next steps	1683
9.13. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE	1683
9.13.1. Prerequisites	1683
9.13.2. About installations in restricted networks	1684
9.13.2.1. Additional limits	1684
9.13.3. Internet access for OpenShift Container Platform	1684

9.13.4. Configuring your GCP project	1685
9.13.4.1. Creating a GCP project	1685
9.13.4.2. Enabling API services in GCP	1685
9.13.4.3. Configuring DNS for GCP	1686
9.13.4.4. GCP account limits	1687
9.13.4.5. Creating a service account in GCP	1688
9.13.4.6. Required GCP roles	1689
9.13.4.7. Required GCP permissions for user-provisioned infrastructure	1690
9.13.4.8. Supported GCP regions	1698
9.13.4.9. Installing and configuring CLI tools for GCP	1699
9.13.5. Requirements for a cluster with user-provisioned infrastructure	1700
9.13.5.1. Required machines for cluster installation	1700
9.13.5.2. Minimum resource requirements for cluster installation	1700
9.13.5.3. Tested instance types for GCP	1701
9.13.5.4. Using custom machine types	1702
9.13.6. Creating the installation files for GCP	1702
9.13.6.1. Optional: Creating a separate /var partition	1702
9.13.6.2. Creating the installation configuration file	1704
9.13.6.3. Configuring the cluster-wide proxy during installation	1707
9.13.6.4. Creating the Kubernetes manifest and Ignition config files	1708
9.13.7. Exporting common variables	1711
9.13.7.1. Extracting the infrastructure name	1711
9.13.7.2. Exporting common variables for Deployment Manager templates	1711
9.13.8. Creating a VPC in GCP	1712
9.13.8.1. Deployment Manager template for the VPC	1713
9.13.9. Networking requirements for user-provisioned infrastructure	1714
9.13.9.1. Setting the cluster node hostnames through DHCP	1714
9.13.9.2. Network connectivity requirements	1714
9.13.10. Creating load balancers in GCP	1716
9.13.10.1. Deployment Manager template for the external load balancer	1717
9.13.10.2. Deployment Manager template for the internal load balancer	1718
9.13.11. Creating a private DNS zone in GCP	1720
9.13.11.1. Deployment Manager template for the private DNS	1721
9.13.12. Creating firewall rules in GCP	1722
9.13.12.1. Deployment Manager template for firewall rules	1723
9.13.13. Creating IAM roles in GCP	1725
9.13.13.1. Deployment Manager template for IAM roles	1727
9.13.14. Creating the RHCOS cluster image for the GCP infrastructure	1727
9.13.15. Creating the bootstrap machine in GCP	1728
9.13.15.1. Deployment Manager template for the bootstrap machine	1730
9.13.16. Creating the control plane machines in GCP	1732
9.13.16.1. Deployment Manager template for control plane machines	1734
9.13.17. Wait for bootstrap completion and remove bootstrap resources in GCP	1736
9.13.18. Creating additional worker machines in GCP	1737
9.13.18.1. Deployment Manager template for worker machines	1739
9.13.19. Logging in to the cluster by using the CLI	1740
9.13.20. Disabling the default OperatorHub catalog sources	1740
9.13.21. Approving the certificate signing requests for your machines	1741
9.13.22. Optional: Adding the ingress DNS records	1744
9.13.23. Completing a GCP installation on user-provisioned infrastructure	1745
9.13.24. Telemetry access for OpenShift Container Platform	1747
9.13.25. Next steps	1748
9.14. UNINSTALLING A CLUSTER ON GCP	1748

9.14.1. Removing a cluster that uses installer-provisioned infrastructure	1748
9.14.2. Deleting GCP resources with the Cloud Credential Operator utility	1749
CHAPTER 10. INSTALLING ON IBM CLOUD VPC	1751
10.1. PREPARING TO INSTALL ON IBM CLOUD VPC	1751
10.1.1. Prerequisites	1751
10.1.2. Requirements for installing OpenShift Container Platform on IBM Cloud VPC	1751
10.1.3. Choosing a method to install OpenShift Container Platform on IBM Cloud VPC	1751
10.1.3.1. Installing a cluster on installer-provisioned infrastructure	1751
10.1.4. Next steps	1752
10.2. CONFIGURING AN IBM CLOUD ACCOUNT	1752
10.2.1. Prerequisites	1752
10.2.2. Quotas and limits on IBM Cloud VPC	1752
Virtual Private Cloud (VPC)	1752
Application load balancer	1752
Floating IP address	1752
Virtual Server Instances (VSI)	1753
Block Storage Volumes	1753
10.2.3. Configuring DNS resolution	1753
10.2.3.1. Using IBM Cloud Internet Services for DNS resolution	1754
10.2.3.2. Using IBM Cloud DNS Services for DNS resolution	1755
10.2.4. IBM Cloud VPC IAM Policies and API Key	1756
10.2.4.1. Required access policies	1756
10.2.4.2. Access policy assignment	1757
10.2.4.3. Creating an API key	1757
10.2.5. Supported IBM Cloud VPC regions	1758
10.2.6. Next steps	1758
10.3. CONFIGURING IAM FOR IBM CLOUD VPC	1758
10.3.1. Alternatives to storing administrator-level secrets in the kube-system project	1758
10.3.2. Configuring the Cloud Credential Operator utility	1759
10.3.3. Next steps	1760
10.3.4. Additional resources	1760
10.4. INSTALLING A CLUSTER ON IBM CLOUD VPC WITH CUSTOMIZATIONS	1760
10.4.1. Prerequisites	1760
10.4.2. Internet access for OpenShift Container Platform	1760
10.4.3. Generating a key pair for cluster node SSH access	1761
10.4.4. Obtaining the installation program	1763
10.4.5. Exporting the IBM Cloud VPC API key	1763
10.4.6. Creating the installation configuration file	1764
10.4.6.1. Installation configuration parameters	1765
10.4.6.1.1. Required configuration parameters	1765
10.4.6.1.2. Network configuration parameters	1767
10.4.6.1.3. Optional configuration parameters	1769
10.4.6.1.4. Additional IBM Cloud VPC configuration parameters	1774
10.4.6.2. Minimum resource requirements for cluster installation	1775
10.4.6.3. Sample customized install-config.yaml file for IBM Cloud VPC	1776
10.4.6.4. Configuring the cluster-wide proxy during installation	1778
10.4.7. Manually creating IAM for IBM Cloud VPC	1779
10.4.8. Deploying the cluster	1782
10.4.9. Installing the OpenShift CLI by downloading the binary	1783
Installing the OpenShift CLI on Linux	1784
Installing the OpenShift CLI on Windows	1784
Installing the OpenShift CLI on macOS	1785

10.4.10. Logging in to the cluster by using the CLI	1785
10.4.11. Telemetry access for OpenShift Container Platform	1786
10.4.12. Next steps	1786
10.5. INSTALLING A CLUSTER ON IBM CLOUD VPC WITH NETWORK CUSTOMIZATIONS	1786
10.5.1. Prerequisites	1786
10.5.2. Internet access for OpenShift Container Platform	1787
10.5.3. Generating a key pair for cluster node SSH access	1787
10.5.4. Obtaining the installation program	1789
10.5.5. Exporting the IBM Cloud VPC API key	1790
10.5.6. Creating the installation configuration file	1790
10.5.6.1. Installation configuration parameters	1791
10.5.6.1.1. Required configuration parameters	1792
10.5.6.1.2. Network configuration parameters	1793
10.5.6.1.3. Optional configuration parameters	1795
10.5.6.1.4. Additional IBM Cloud VPC configuration parameters	1800
10.5.6.2. Minimum resource requirements for cluster installation	1801
10.5.6.3. Sample customized install-config.yaml file for IBM Cloud VPC	1802
10.5.6.4. Configuring the cluster-wide proxy during installation	1804
10.5.7. Manually creating IAM for IBM Cloud VPC	1805
10.5.8. Network configuration phases	1808
10.5.9. Specifying advanced network configuration	1809
10.5.10. Cluster Network Operator configuration	1809
10.5.10.1. Cluster Network Operator configuration object	1810
defaultNetwork object configuration	1811
Configuration for the OpenShift SDN network plugin	1811
Configuration for the OVN-Kubernetes network plugin	1812
kubeProxyConfig object configuration	1816
10.5.11. Deploying the cluster	1817
10.5.12. Installing the OpenShift CLI by downloading the binary	1819
Installing the OpenShift CLI on Linux	1819
Installing the OpenShift CLI on Windows	1820
Installing the OpenShift CLI on macOS	1820
10.5.13. Logging in to the cluster by using the CLI	1821
10.5.14. Telemetry access for OpenShift Container Platform	1821
10.5.15. Next steps	1822
10.6. INSTALLING A CLUSTER ON IBM CLOUD VPC INTO AN EXISTING VPC	1822
10.6.1. Prerequisites	1822
10.6.2. About using a custom VPC	1822
10.6.2.1. Requirements for using your VPC	1822
10.6.2.2. VPC validation	1823
10.6.2.3. Isolation between clusters	1823
10.6.3. Internet access for OpenShift Container Platform	1824
10.6.4. Generating a key pair for cluster node SSH access	1824
10.6.5. Obtaining the installation program	1826
10.6.6. Exporting the IBM Cloud VPC API key	1827
10.6.7. Creating the installation configuration file	1827
10.6.7.1. Installation configuration parameters	1828
10.6.7.1.1. Required configuration parameters	1828
10.6.7.1.2. Network configuration parameters	1830
10.6.7.1.3. Optional configuration parameters	1832
10.6.7.1.4. Additional IBM Cloud VPC configuration parameters	1837
10.6.7.2. Minimum resource requirements for cluster installation	1838
10.6.7.3. Sample customized install-config.yaml file for IBM Cloud VPC	1839

10.6.7.4. Configuring the cluster-wide proxy during installation	1841
10.6.8. Manually creating IAM for IBM Cloud VPC	1843
10.6.9. Deploying the cluster	1845
10.6.10. Installing the OpenShift CLI by downloading the binary	1847
Installing the OpenShift CLI on Linux	1847
Installing the OpenShift CLI on Windows	1848
Installing the OpenShift CLI on macOS	1848
10.6.11. Logging in to the cluster by using the CLI	1849
10.6.12. Telemetry access for OpenShift Container Platform	1849
10.6.13. Next steps	1850
10.7. INSTALLING A PRIVATE CLUSTER ON IBM CLOUD VPC	1850
10.7.1. Prerequisites	1850
10.7.2. Private clusters	1850
10.7.3. Private clusters in IBM Cloud VPC	1851
10.7.3.1. Limitations	1851
10.7.4. About using a custom VPC	1851
10.7.4.1. Requirements for using your VPC	1851
10.7.4.2. VPC validation	1852
10.7.4.3. Isolation between clusters	1852
10.7.5. Internet access for OpenShift Container Platform	1853
10.7.6. Generating a key pair for cluster node SSH access	1853
10.7.7. Obtaining the installation program	1855
10.7.8. Exporting the IBM Cloud VPC API key	1856
10.7.9. Manually creating the installation configuration file	1856
10.7.9.1. Installation configuration parameters	1857
10.7.9.1.1. Required configuration parameters	1857
10.7.9.1.2. Network configuration parameters	1859
10.7.9.1.3. Optional configuration parameters	1861
10.7.9.1.4. Additional IBM Cloud VPC configuration parameters	1866
10.7.9.2. Minimum resource requirements for cluster installation	1867
10.7.9.3. Sample customized install-config.yaml file for IBM Cloud VPC	1868
10.7.9.4. Configuring the cluster-wide proxy during installation	1870
10.7.10. Manually creating IAM for IBM Cloud VPC	1872
10.7.11. Deploying the cluster	1874
10.7.12. Installing the OpenShift CLI by downloading the binary	1876
Installing the OpenShift CLI on Linux	1876
Installing the OpenShift CLI on Windows	1877
Installing the OpenShift CLI on macOS	1877
10.7.13. Logging in to the cluster by using the CLI	1878
10.7.14. Telemetry access for OpenShift Container Platform	1878
10.7.15. Next steps	1879
10.8. UNINSTALLING A CLUSTER ON IBM CLOUD VPC	1879
10.8.1. Removing a cluster that uses installer-provisioned infrastructure	1879
CHAPTER 11. INSTALLING ON NUTANIX	1881
11.1. PREPARING TO INSTALL ON NUTANIX	1881
11.1.1. Nutanix version requirements	1881
11.1.2. Environment requirements	1881
11.1.2.1. Required account privileges	1881
11.1.2.2. Cluster limits	1882
11.1.2.3. Cluster resources	1882
11.1.2.4. Networking requirements	1883
11.1.2.4.1. Required IP Addresses	1883

11.1.2.4.2. DNS records	1883
11.1.3. Configuring the Cloud Credential Operator utility	1884
11.2. INSTALLING A CLUSTER ON NUTANIX	1885
11.2.1. Prerequisites	1886
11.2.2. Internet access for OpenShift Container Platform	1886
11.2.3. Internet access for Prism Central	1887
11.2.4. Generating a key pair for cluster node SSH access	1887
11.2.5. Obtaining the installation program	1889
11.2.6. Adding Nutanix root CA certificates to your system trust	1889
11.2.7. Creating the installation configuration file	1890
11.2.7.1. Installation configuration parameters	1891
11.2.7.1.1. Required configuration parameters	1892
11.2.7.1.2. Network configuration parameters	1893
11.2.7.1.3. Optional configuration parameters	1895
11.2.7.1.4. Additional Nutanix configuration parameters	1900
11.2.7.2. Sample customized install-config.yaml file for Nutanix	1902
11.2.7.3. Configuring the cluster-wide proxy during installation	1904
11.2.8. Installing the OpenShift CLI by downloading the binary	1905
Installing the OpenShift CLI on Linux	1906
Installing the OpenShift CLI on Windows	1906
Installing the OpenShift CLI on macOS	1907
11.2.9. Configuring IAM for Nutanix	1907
11.2.10. Deploying the cluster	1910
11.2.11. Configuring the default storage container	1912
11.2.12. Telemetry access for OpenShift Container Platform	1912
11.2.13. Additional resources	1912
11.2.14. Next steps	1912
11.3. INSTALLING A CLUSTER ON NUTANIX IN A RESTRICTED NETWORK	1912
11.3.1. Prerequisites	1912
11.3.2. About installations in restricted networks	1913
11.3.2.1. Additional limits	1914
11.3.3. Generating a key pair for cluster node SSH access	1914
11.3.4. Adding Nutanix root CA certificates to your system trust	1915
11.3.5. Downloading the RHCOS cluster image	1916
11.3.6. Creating the installation configuration file	1917
11.3.6.1. Installation configuration parameters	1919
11.3.6.1.1. Required configuration parameters	1919
11.3.6.1.2. Network configuration parameters	1921
11.3.6.1.3. Optional configuration parameters	1923
11.3.6.1.4. Additional Nutanix configuration parameters	1928
11.3.6.2. Sample customized install-config.yaml file for Nutanix	1930
11.3.6.3. Configuring the cluster-wide proxy during installation	1932
11.3.7. Installing the OpenShift CLI by downloading the binary	1934
Installing the OpenShift CLI on Linux	1934
Installing the OpenShift CLI on Windows	1935
Installing the OpenShift CLI on macOS	1935
11.3.8. Configuring IAM for Nutanix	1936
11.3.9. Deploying the cluster	1938
11.3.10. Post installation	1940
11.3.10.1. Disabling the default OperatorHub catalog sources	1940
11.3.10.2. Installing the policy resources into the cluster	1940
11.3.10.3. Configuring the default storage container	1941
11.3.11. Telemetry access for OpenShift Container Platform	1941

11.3.12. Additional resources	1942
11.3.13. Next steps	1942
11.4. UNINSTALLING A CLUSTER ON NUTANIX	1942
11.4.1. Removing a cluster that uses installer-provisioned infrastructure	1942
CHAPTER 12. INSTALLING ON BARE METAL	1944
12.1. PREPARING FOR BARE METAL CLUSTER INSTALLATION	1944
12.1.1. Prerequisites	1944
12.1.2. Planning a bare metal cluster for OpenShift Virtualization	1944
12.1.3. Choosing a method to install OpenShift Container Platform on bare metal	1944
12.1.3.1. Installing a cluster on installer-provisioned infrastructure	1945
12.1.3.2. Installing a cluster on user-provisioned infrastructure	1945
12.2. INSTALLING A USER-PROVISIONED CLUSTER ON BARE METAL	1945
12.2.1. Prerequisites	1946
12.2.2. Internet access for OpenShift Container Platform	1946
12.2.3. Requirements for a cluster with user-provisioned infrastructure	1946
12.2.3.1. Required machines for cluster installation	1947
12.2.3.2. Minimum resource requirements for cluster installation	1947
12.2.3.3. Certificate signing requests management	1948
12.2.3.4. Requirements for baremetal clusters on vSphere	1949
12.2.3.5. Networking requirements for user-provisioned infrastructure	1949
12.2.3.5.1. Setting the cluster node hostnames through DHCP	1949
12.2.3.5.2. Network connectivity requirements	1949
NTP configuration for user-provisioned infrastructure	1951
12.2.3.6. User-provisioned DNS requirements	1951
12.2.3.6.1. Example DNS configuration for user-provisioned clusters	1953
12.2.3.7. Load balancing requirements for user-provisioned infrastructure	1955
12.2.3.7.1. Example load balancer configuration for user-provisioned clusters	1957
12.2.4. Preparing the user-provisioned infrastructure	1959
12.2.5. Validating DNS resolution for user-provisioned infrastructure	1961
12.2.6. Generating a key pair for cluster node SSH access	1963
12.2.7. Obtaining the installation program	1965
12.2.8. Installing the OpenShift CLI by downloading the binary	1966
Installing the OpenShift CLI on Linux	1966
Installing the OpenShift CLI on Windows	1967
Installing the OpenShift CLI on macOS	1967
12.2.9. Manually creating the installation configuration file	1968
12.2.9.1. Installation configuration parameters	1969
12.2.9.1.1. Required configuration parameters	1969
12.2.9.1.2. Network configuration parameters	1970
12.2.9.1.3. Optional configuration parameters	1973
12.2.9.2. Sample install-config.yaml file for bare metal	1978
12.2.9.3. Configuring the cluster-wide proxy during installation	1981
12.2.9.4. Configuring a three-node cluster	1983
12.2.10. Creating the Kubernetes manifest and Ignition config files	1984
12.2.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	1985
12.2.11.1. Installing RHCOS by using an ISO image	1986
12.2.11.2. Installing RHCOS by using PXE or iPXE booting	1990
12.2.11.3. Advanced RHCOS installation configuration	1994
12.2.11.3.1. Using advanced networking options for PXE and ISO installations	1994
12.2.11.3.2. Disk partitioning	1995
12.2.11.3.2.1. Creating a separate /var partition	1996
12.2.11.3.2.2. Retaining existing partitions	1998

12.2.11.3.3. Identifying Ignition configs	1999
12.2.11.3.4. Default console configuration	2000
12.2.11.3.5. Enabling the serial console for PXE and ISO installations	2000
12.2.11.3.6. Customizing a live RHCOS ISO or PXE install	2001
12.2.11.3.7. Customizing a live RHCOS ISO image	2002
12.2.11.3.7.1. Modifying a live install ISO image to enable the serial console	2002
12.2.11.3.7.2. Modifying a live install ISO image to use a custom certificate authority	2003
12.2.11.3.7.3. Modifying a live install ISO image with customized network settings	2004
12.2.11.3.8. Customizing a live RHCOS PXE environment	2005
12.2.11.3.8.1. Modifying a live install PXE environment to enable the serial console	2006
12.2.11.3.8.2. Modifying a live install PXE environment to use a custom certificate authority	2006
12.2.11.3.8.3. Modifying a live install PXE environment with customized network settings	2007
12.2.11.3.9. Advanced RHCOS installation reference	2009
12.2.11.3.9.1. Networking and bonding options for ISO installations	2009
Configuring DHCP or static IP addresses	2009
Configuring an IP address without a static hostname	2010
Specifying multiple network interfaces	2010
Configuring default gateway and route	2010
Disabling DHCP on a single interface	2011
Combining DHCP and static IP configurations	2011
Configuring VLANs on individual interfaces	2011
Providing multiple DNS servers	2011
Bonding multiple network interfaces to a single interface	2011
Bonding multiple network interfaces to a single interface	2012
Using network teaming	2012
12.2.11.3.9.2. coreos-installer options for ISO and PXE installations	2012
12.2.11.3.9.3. coreos.inst boot options for ISO or PXE installations	2016
12.2.11.4. Enabling multipathing with kernel arguments on RHCOS	2018
12.2.12. Waiting for the bootstrap process to complete	2020
12.2.13. Logging in to the cluster by using the CLI	2021
12.2.14. Approving the certificate signing requests for your machines	2021
12.2.15. Initial Operator configuration	2024
12.2.15.1. Image registry removed during installation	2025
12.2.15.2. Image registry storage configuration	2025
12.2.15.2.1. Configuring registry storage for bare metal and other manual installations	2026
12.2.15.2.2. Configuring storage for the image registry in non-production clusters	2027
12.2.15.2.3. Configuring block registry storage for bare metal	2028
12.2.16. Completing installation on user-provisioned infrastructure	2029
12.2.17. Telemetry access for OpenShift Container Platform	2032
12.2.18. Next steps	2032
12.3. INSTALLING A USER-PROVISIONED BARE METAL CLUSTER WITH NETWORK CUSTOMIZATIONS	2032
12.3.1. Prerequisites	2032
12.3.2. Internet access for OpenShift Container Platform	2032
12.3.3. Requirements for a cluster with user-provisioned infrastructure	2033
12.3.3.1. Required machines for cluster installation	2033
12.3.3.2. Minimum resource requirements for cluster installation	2034
12.3.3.3. Certificate signing requests management	2035
12.3.3.4. Networking requirements for user-provisioned infrastructure	2035
12.3.3.4.1. Setting the cluster node hostnames through DHCP	2036
12.3.3.4.2. Network connectivity requirements	2036
NTP configuration for user-provisioned infrastructure	2037
12.3.3.5. User-provisioned DNS requirements	2037
12.3.3.5.1. Example DNS configuration for user-provisioned clusters	2039

12.3.3.6. Load balancing requirements for user-provisioned infrastructure	2041
12.3.3.6.1. Example load balancer configuration for user-provisioned clusters	2043
12.3.4. Preparing the user-provisioned infrastructure	2045
12.3.5. Validating DNS resolution for user-provisioned infrastructure	2047
12.3.6. Generating a key pair for cluster node SSH access	2050
12.3.7. Obtaining the installation program	2051
12.3.8. Installing the OpenShift CLI by downloading the binary	2052
Installing the OpenShift CLI on Linux	2052
Installing the OpenShift CLI on Windows	2053
Installing the OpenShift CLI on macOS	2053
12.3.9. Manually creating the installation configuration file	2054
12.3.9.1. Installation configuration parameters	2055
12.3.9.1.1. Required configuration parameters	2055
12.3.9.1.2. Network configuration parameters	2056
12.3.9.1.3. Optional configuration parameters	2059
12.3.9.2. Sample install-config.yaml file for bare metal	2064
12.3.10. Network configuration phases	2067
12.3.11. Specifying advanced network configuration	2067
12.3.12. Cluster Network Operator configuration	2069
12.3.12.1. Cluster Network Operator configuration object	2069
defaultNetwork object configuration	2070
Configuration for the OpenShift SDN network plugin	2070
Configuration for the OVN-Kubernetes network plugin	2071
kubeProxyConfig object configuration	2075
12.3.13. Creating the Ignition config files	2076
12.3.14. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2077
12.3.14.1. Installing RHCOS by using an ISO image	2078
12.3.14.2. Installing RHCOS by using PXE or iPXE booting	2081
12.3.14.3. Advanced RHCOS installation configuration	2086
12.3.14.3.1. Using advanced networking options for PXE and ISO installations	2086
12.3.14.3.2. Disk partitioning	2087
12.3.14.3.2.1. Creating a separate /var partition	2088
12.3.14.3.2.2. Retaining existing partitions	2090
12.3.14.3.3. Identifying Ignition configs	2091
12.3.14.3.4. Default console configuration	2092
12.3.14.3.5. Enabling the serial console for PXE and ISO installations	2092
12.3.14.3.6. Customizing a live RHCOS ISO or PXE install	2093
12.3.14.3.7. Customizing a live RHCOS ISO image	2094
12.3.14.3.7.1. Modifying a live install ISO image to enable the serial console	2094
12.3.14.3.7.2. Modifying a live install ISO image to use a custom certificate authority	2095
12.3.14.3.7.3. Modifying a live install ISO image with customized network settings	2096
12.3.14.3.8. Customizing a live RHCOS PXE environment	2097
12.3.14.3.8.1. Modifying a live install PXE environment to enable the serial console	2098
12.3.14.3.8.2. Modifying a live install PXE environment to use a custom certificate authority	2098
12.3.14.3.8.3. Modifying a live install PXE environment with customized network settings	2099
12.3.14.3.9. Advanced RHCOS installation reference	2101
12.3.14.3.9.1. Networking and bonding options for ISO installations	2101
Configuring DHCP or static IP addresses	2101
Configuring an IP address without a static hostname	2102
Specifying multiple network interfaces	2102
Configuring default gateway and route	2102
Disabling DHCP on a single interface	2103
Combining DHCP and static IP configurations	2103

Configuring VLANs on individual interfaces	2103
Providing multiple DNS servers	2103
Bonding multiple network interfaces to a single interface	2103
Bonding multiple network interfaces to a single interface	2104
Using network teaming	2104
12.3.14.3.9.2. coreos-installer options for ISO and PXE installations	2104
12.3.14.3.9.3. coreos.inst boot options for ISO or PXE installations	2108
12.3.14.4. Enabling multipathing with kernel arguments on RHCOS	2110
12.3.15. Waiting for the bootstrap process to complete	2112
12.3.16. Logging in to the cluster by using the CLI	2113
12.3.17. Approving the certificate signing requests for your machines	2113
12.3.18. Initial Operator configuration	2116
12.3.18.1. Image registry removed during installation	2117
12.3.18.2. Image registry storage configuration	2117
12.3.18.3. Configuring block registry storage for bare metal	2118
12.3.19. Completing installation on user-provisioned infrastructure	2119
12.3.20. Telemetry access for OpenShift Container Platform	2121
12.3.21. Next steps	2121
12.4. INSTALLING A USER-PROVISIONED BARE METAL CLUSTER ON A RESTRICTED NETWORK	2122
12.4.1. Prerequisites	2122
12.4.2. About installations in restricted networks	2122
12.4.2.1. Additional limits	2123
12.4.3. Internet access for OpenShift Container Platform	2123
12.4.4. Requirements for a cluster with user-provisioned infrastructure	2123
12.4.4.1. Required machines for cluster installation	2124
12.4.4.2. Minimum resource requirements for cluster installation	2124
12.4.4.3. Certificate signing requests management	2125
12.4.4.4. Networking requirements for user-provisioned infrastructure	2126
12.4.4.4.1. Setting the cluster node hostnames through DHCP	2126
12.4.4.4.2. Network connectivity requirements	2126
NTP configuration for user-provisioned infrastructure	2127
12.4.4.5. User-provisioned DNS requirements	2128
12.4.4.5.1. Example DNS configuration for user-provisioned clusters	2129
12.4.4.6. Load balancing requirements for user-provisioned infrastructure	2132
12.4.4.6.1. Example load balancer configuration for user-provisioned clusters	2134
12.4.5. Preparing the user-provisioned infrastructure	2136
12.4.6. Validating DNS resolution for user-provisioned infrastructure	2138
12.4.7. Generating a key pair for cluster node SSH access	2140
12.4.8. Manually creating the installation configuration file	2142
12.4.8.1. Installation configuration parameters	2143
12.4.8.1.1. Required configuration parameters	2143
12.4.8.1.2. Network configuration parameters	2145
12.4.8.1.3. Optional configuration parameters	2147
12.4.8.2. Sample install-config.yaml file for bare metal	2153
12.4.8.3. Configuring the cluster-wide proxy during installation	2156
12.4.8.4. Configuring a three-node cluster	2158
12.4.9. Creating the Kubernetes manifest and Ignition config files	2159
12.4.10. Configuring chrony time service	2161
12.4.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2162
12.4.11.1. Installing RHCOS by using an ISO image	2163
12.4.11.2. Installing RHCOS by using PXE or iPXE booting	2166
12.4.11.3. Advanced RHCOS installation configuration	2171
12.4.11.3.1. Using advanced networking options for PXE and ISO installations	2171

12.4.11.3.2. Disk partitioning	2172
12.4.11.3.2.1. Creating a separate /var partition	2173
12.4.11.3.2.2. Retaining existing partitions	2175
12.4.11.3.3. Identifying Ignition configs	2176
12.4.11.3.4. Default console configuration	2176
12.4.11.3.5. Enabling the serial console for PXE and ISO installations	2177
12.4.11.3.6. Customizing a live RHCOS ISO or PXE install	2178
12.4.11.3.7. Customizing a live RHCOS ISO image	2178
12.4.11.3.7.1. Modifying a live install ISO image to enable the serial console	2179
12.4.11.3.7.2. Modifying a live install ISO image to use a custom certificate authority	2180
12.4.11.3.7.3. Modifying a live install ISO image with customized network settings	2180
12.4.11.3.8. Customizing a live RHCOS PXE environment	2182
12.4.11.3.8.1. Modifying a live install PXE environment to enable the serial console	2182
12.4.11.3.8.2. Modifying a live install PXE environment to use a custom certificate authority	2183
12.4.11.3.8.3. Modifying a live install PXE environment with customized network settings	2184
12.4.11.3.9. Advanced RHCOS installation reference	2185
12.4.11.3.9.1. Networking and bonding options for ISO installations	2185
Configuring DHCP or static IP addresses	2186
Configuring an IP address without a static hostname	2186
Specifying multiple network interfaces	2187
Configuring default gateway and route	2187
Disabling DHCP on a single interface	2187
Combining DHCP and static IP configurations	2187
Configuring VLANs on individual interfaces	2188
Providing multiple DNS servers	2188
Bonding multiple network interfaces to a single interface	2188
Bonding multiple network interfaces to a single interface	2188
Using network teaming	2189
12.4.11.3.9.2. coreos-installer options for ISO and PXE installations	2189
12.4.11.3.9.3. coreos.inst boot options for ISO or PXE installations	2193
12.4.11.4. Enabling multipathing with kernel arguments on RHCOS	2195
12.4.12. Waiting for the bootstrap process to complete	2196
12.4.13. Logging in to the cluster by using the CLI	2197
12.4.14. Approving the certificate signing requests for your machines	2198
12.4.15. Initial Operator configuration	2201
12.4.15.1. Disabling the default OperatorHub catalog sources	2202
12.4.15.2. Image registry storage configuration	2202
12.4.15.2.1. Changing the image registry's management state	2202
12.4.15.2.2. Configuring registry storage for bare metal and other manual installations	2203
12.4.15.2.3. Configuring storage for the image registry in non-production clusters	2204
12.4.15.2.4. Configuring block registry storage for bare metal	2205
12.4.16. Completing installation on user-provisioned infrastructure	2206
12.4.17. Telemetry access for OpenShift Container Platform	2209
12.4.18. Next steps	2209
CHAPTER 13. INSTALLING ON-PREMISE WITH ASSISTED INSTALLER	2210
13.1. INSTALLING AN ON-PREMISE CLUSTER USING THE ASSISTED INSTALLER	2210
13.1.1. Using the Assisted Installer	2210
13.1.2. API support for the Assisted Installer	2211
CHAPTER 14. INSTALLING AN ON-PREMISE CLUSTER WITH THE AGENT-BASED INSTALLER	2212
14.1. PREPARING TO INSTALL WITH THE AGENT-BASED INSTALLER	2212
14.1.1. About the Agent-based Installer	2212

14.1.2. Understanding Agent-based Installer	2212
14.1.2.1. Agent-based Installer workflow	2213
14.1.2.2. Recommended resources for topologies	2214
14.1.3. About FIPS compliance	2214
14.1.4. Configuring FIPS through the Agent-based Installer	2215
14.1.5. About networking	2215
14.1.5.1. DHCP	2215
14.1.5.2. Static networking	2216
14.1.6. Example: Bonds and VLAN interface node network configuration	2218
14.1.7. Sample install-config.yaml file for bare metal	2219
14.1.8. Validation checks before agent ISO creation	2222
14.1.8.1. ZTP manifests	2222
14.1.9. About root device hints	2222
14.1.10. Next steps	2223
14.2. UNDERSTANDING DISCONNECTED INSTALLATION MIRRORING	2223
14.2.1. Mirroring images for a disconnected installation through the Agent-based Installer	2224
14.2.2. About mirroring the OpenShift Container Platform image repository for a disconnected registry	2224
14.2.2.1. Configuring the Agent-based Installer to use mirrored images	2225
14.3. INSTALLING AN OPENSIFT CONTAINER PLATFORM CLUSTER WITH THE AGENT-BASED INSTALLER	2226
14.3.1. Prerequisites	2226
14.3.2. Installing OpenShift Container Platform with the Agent-based Installer	2226
14.3.3. Gathering log data from a failed Agent-based installation	2232
14.3.4. Sample ZTP custom resources	2233
14.4. PREPARING AN AGENT-BASED INSTALLED CLUSTER FOR THE MULTICLUSTER ENGINE FOR KUBERNETES OPERATOR	2236
14.4.1. Prerequisites	2236
14.4.2. Preparing an Agent-based cluster deployment for the multicluster engine for Kubernetes Operator while disconnected	2236
14.4.3. Preparing an Agent-based cluster deployment for the multicluster engine for Kubernetes Operator while connected	2239
CHAPTER 15. INSTALLING ON A SINGLE NODE	2244
15.1. PREPARING TO INSTALL ON A SINGLE NODE	2244
15.1.1. Prerequisites	2244
15.1.2. About OpenShift on a single node	2244
15.1.3. Requirements for installing OpenShift on a single node	2244
15.2. INSTALLING OPENSIFT ON A SINGLE NODE	2245
15.2.1. Installing single-node OpenShift using the Assisted Installer	2245
15.2.1.1. Generating the discovery ISO with the Assisted Installer	2246
15.2.1.2. Installing single-node OpenShift with the Assisted Installer	2246
15.2.2. Installing single-node OpenShift manually	2247
15.2.2.1. Generating the installation ISO with coreos-installer	2247
15.2.2.2. Monitoring the cluster installation using openshift-install	2250
15.2.3. Creating a bootable ISO image on a USB drive	2250
15.2.4. Booting from an HTTP-hosted ISO image using the Redfish API	2251
15.2.5. Creating a custom live RHCOS ISO for remote server access	2252
CHAPTER 16. DEPLOYING INSTALLER-PROVISIONED CLUSTERS ON BARE METAL	2254
16.1. OVERVIEW	2254
16.2. PREREQUISITES	2255
16.2.1. Node requirements	2256
16.2.2. Planning a bare metal cluster for OpenShift Virtualization	2257
16.2.3. Firmware requirements for installing with virtual media	2258

16.2.4. Network requirements	2258
16.2.4.1. Increase the network MTU	2259
16.2.4.2. Configuring NICs	2259
16.2.4.3. DNS requirements	2260
16.2.4.4. Dynamic Host Configuration Protocol (DHCP) requirements	2261
16.2.4.5. Reserving IP addresses for nodes with the DHCP server	2261
16.2.4.6. Provisioner node requirements	2262
16.2.4.7. Network Time Protocol (NTP)	2262
16.2.4.8. Port access for the out-of-band management IP address	2263
16.2.5. Configuring nodes	2263
Configuring nodes when using the provisioning network	2263
Configuring nodes without the provisioning network	2264
Configuring nodes for Secure Boot manually	2264
16.2.6. Out-of-band management	2265
16.2.7. Required data for installation	2265
16.2.8. Validation checklist for nodes	2265
16.3. SETTING UP THE ENVIRONMENT FOR AN OPENSIFT INSTALLATION	2266
16.3.1. Installing RHEL on the provisioner node	2266
16.3.2. Preparing the provisioner node for OpenShift Container Platform installation	2266
16.3.3. Checking NTP server synchronization	2268
16.3.4. Configuring networking	2269
16.3.5. Establishing communication between subnets	2271
16.3.6. Retrieving the OpenShift Container Platform installer	2273
16.3.7. Extracting the OpenShift Container Platform installer	2273
16.3.8. Optional: Creating an RHCOS images cache	2274
16.3.9. Setting the cluster node hostnames through DHCP	2276
16.3.10. Configuring the install-config.yaml file	2276
16.3.10.1. Configuring the install-config.yaml file	2276
16.3.10.2. Additional install-config parameters	2279
Hosts	2283
16.3.10.3. BMC addressing	2284
IPMI	2284
Redfish network boot	2285
Redfish APIs	2285
16.3.10.4. BMC addressing for Dell iDRAC	2286
BMC address formats for Dell iDRAC	2287
Redfish virtual media for Dell iDRAC	2287
Redfish network boot for iDRAC	2288
16.3.10.5. BMC addressing for HPE iLO	2289
Redfish virtual media for HPE iLO	2290
Redfish network boot for HPE iLO	2290
16.3.10.6. BMC addressing for Fujitsu iRMC	2291
16.3.10.7. Root device hints	2292
16.3.10.8. Optional: Setting proxy settings	2293
16.3.10.9. Optional: Deploying with no provisioning network	2294
16.3.10.10. Optional: Deploying with dual-stack networking	2294
16.3.10.11. Optional: Configuring host network interfaces	2295
16.3.10.12. Configuring host network interfaces for subnets	2297
16.3.10.13. Optional: Configuring address generation modes for SLAAC in dual-stack networks	2299
16.3.10.14. Configuring multiple cluster nodes	2300
16.3.10.15. Optional: Configuring managed Secure Boot	2300
16.3.11. Manifest configuration files	2301
16.3.11.1. Creating the OpenShift Container Platform manifests	2301

16.3.11.2. Optional: Configuring NTP for disconnected clusters	2301
16.3.11.3. Configuring network components to run on the control plane	2304
16.3.11.4. Optional: Deploying routers on worker nodes	2306
16.3.11.5. Optional: Configuring the BIOS	2307
16.3.11.6. Optional: Configuring the RAID	2308
16.3.12. Creating a disconnected registry	2309
Prerequisites	2309
16.3.12.1. Preparing the registry node to host the mirrored registry	2309
16.3.12.2. Mirroring the OpenShift Container Platform image repository for a disconnected registry	2310
16.3.12.3. Modify the install-config.yaml file to use the disconnected registry	2313
16.3.13. Validation checklist for installation	2314
16.3.14. Deploying the cluster via the OpenShift Container Platform installer	2314
16.3.15. Following the installation	2314
16.3.16. Verifying static IP address configuration	2314
16.3.17. Preparing to reinstall a cluster on bare metal	2315
16.3.18. Additional resources	2315
16.4. INSTALLER-PROVISIONED POSTINSTALLATION CONFIGURATION	2315
16.4.1. Optional: Configuring NTP for disconnected clusters	2315
16.4.2. Enabling a provisioning network after installation	2318
16.4.3. Services for an external load balancer	2320
16.4.3.1. Configuring an external load balancer	2323
16.5. EXPANDING THE CLUSTER	2328
16.5.1. Preparing the bare metal node	2329
16.5.2. Replacing a bare-metal control plane node	2333
16.5.3. Preparing to deploy with Virtual Media on the baremetal network	2337
16.5.4. Diagnosing a duplicate MAC address when provisioning a new host in the cluster	2339
16.5.5. Provisioning the bare metal node	2340
16.6. TROUBLESHOOTING	2342
16.6.1. Troubleshooting the installer workflow	2342
16.6.2. Troubleshooting install-config.yaml	2344
16.6.3. Bootstrap VM issues	2345
16.6.3.1. Bootstrap VM cannot boot up the cluster nodes	2346
16.6.3.2. Inspecting logs	2347
16.6.4. Cluster nodes will not PXE boot	2348
16.6.5. Unable to discover new bare metal hosts using the BMC	2348
16.6.6. The API is not accessible	2349
16.6.7. Troubleshooting worker nodes that cannot join the cluster	2350
16.6.8. Cleaning up previous installations	2351
16.6.9. Issues with creating the registry	2351
16.6.10. Miscellaneous issues	2352
16.6.10.1. Addressing the runtime network not ready error	2352
16.6.10.2. Addressing the "No disk found with matching rootDeviceHints" error message	2353
16.6.10.3. Cluster nodes not getting the correct IPv6 address over DHCP	2353
16.6.10.4. Cluster nodes not getting the correct hostname over DHCP	2354
16.6.10.5. Routes do not reach endpoints	2355
16.6.10.6. Failed Ignition during Firstboot	2356
16.6.10.7. NTP out of sync	2357
16.6.11. Reviewing the installation	2359
CHAPTER 17. INSTALLING IBM CLOUD BARE METAL (CLASSIC)	2360
17.1. PREREQUISITES	2360
17.1.1. Setting up IBM Cloud Bare Metal (Classic) infrastructure	2360
Use one data center per cluster	2360

Create public and private VLANs	2360
Ensure subnets have sufficient IP addresses	2360
Configuring NICs	2361
Configuring canonical names	2362
Creating DNS entries	2362
Network Time Protocol (NTP)	2363
Configure a DHCP server	2363
Ensure BMC access privileges	2363
Create bare metal servers	2364
17.2. SETTING UP THE ENVIRONMENT FOR AN OPENSIFT CONTAINER PLATFORM INSTALLATION	2364
17.2.1. Preparing the provisioner node on IBM Cloud Bare Metal (Classic) infrastructure	2364
17.2.2. Configuring the public subnet	2368
17.2.3. Retrieving the OpenShift Container Platform installer	2370
17.2.4. Extracting the OpenShift Container Platform installer	2371
17.2.5. Configuring the install-config.yaml file	2371
17.2.6. Additional install-config parameters	2373
Hosts	2377
17.2.7. Root device hints	2378
17.2.8. Creating the OpenShift Container Platform manifests	2379
17.2.9. Deploying the cluster via the OpenShift Container Platform installer	2379
17.2.10. Following the installation	2379
CHAPTER 18. INSTALLING WITH Z/VM ON IBM Z AND IBM LINUXONE	2381
18.1. PREPARING TO INSTALL WITH Z/VM ON IBM Z AND IBM(R) LINUXONE	2381
18.1.1. Prerequisites	2381
18.1.2. Choosing a method to install OpenShift Container Platform with z/VM on IBM Z or IBM(R) LinuxONE	2381
18.2. INSTALLING A CLUSTER WITH Z/VM ON IBM Z AND IBM(R) LINUXONE	2381
18.2.1. Prerequisites	2381
18.2.2. Internet access for OpenShift Container Platform	2382
18.2.3. Requirements for a cluster with user-provisioned infrastructure	2382
18.2.3.1. Required machines for cluster installation	2382
18.2.3.2. Minimum resource requirements for cluster installation	2383
18.2.3.3. Minimum IBM Z system environment	2384
Hardware requirements	2384
Operating system requirements	2384
IBM Z network connectivity requirements	2384
Disk storage for the z/VM guest virtual machines	2385
Storage / Main Memory	2385
18.2.3.4. Preferred IBM Z system environment	2385
Hardware requirements	2385
Operating system requirements	2385
IBM Z network connectivity requirements	2385
Disk storage for the z/VM guest virtual machines	2386
Storage / Main Memory	2386
18.2.3.5. Certificate signing requests management	2386
18.2.3.6. Networking requirements for user-provisioned infrastructure	2386
18.2.3.6.1. Network connectivity requirements	2387
NTP configuration for user-provisioned infrastructure	2388
18.2.3.7. User-provisioned DNS requirements	2388
18.2.3.7.1. Example DNS configuration for user-provisioned clusters	2390
18.2.3.8. Load balancing requirements for user-provisioned infrastructure	2392
18.2.3.8.1. Example load balancer configuration for user-provisioned clusters	2394

18.2.4. Preparing the user-provisioned infrastructure	2396
18.2.5. Validating DNS resolution for user-provisioned infrastructure	2397
18.2.6. Generating a key pair for cluster node SSH access	2400
18.2.7. Obtaining the installation program	2401
18.2.8. Installing the OpenShift CLI by downloading the binary	2402
Installing the OpenShift CLI on Linux	2402
Installing the OpenShift CLI on Windows	2403
Installing the OpenShift CLI on macOS	2403
18.2.9. Manually creating the installation configuration file	2404
18.2.9.1. Installation configuration parameters	2405
18.2.9.1.1. Required configuration parameters	2405
18.2.9.1.2. Network configuration parameters	2406
18.2.9.1.3. Optional configuration parameters	2408
18.2.9.2. Sample install-config.yaml file for IBM Z	2414
18.2.9.3. Configuring the cluster-wide proxy during installation	2417
18.2.9.4. Configuring a three-node cluster	2419
18.2.10. Cluster Network Operator configuration	2420
18.2.10.1. Cluster Network Operator configuration object	2420
defaultNetwork object configuration	2421
Configuration for the OpenShift SDN network plugin	2421
Configuration for the OVN-Kubernetes network plugin	2422
kubeProxyConfig object configuration	2426
18.2.11. Creating the Kubernetes manifest and Ignition config files	2427
18.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2429
18.2.12.1. Advanced RHCOS installation reference	2432
18.2.12.1.1. Networking and bonding options for ISO installations	2432
Configuring DHCP or static IP addresses	2432
Configuring an IP address without a static hostname	2433
Specifying multiple network interfaces	2433
Configuring default gateway and route	2433
Disabling DHCP on a single interface	2434
Combining DHCP and static IP configurations	2434
Configuring VLANs on individual interfaces	2434
Providing multiple DNS servers	2434
Bonding multiple network interfaces to a single interface	2434
Bonding multiple network interfaces to a single interface	2435
Using network teaming	2435
18.2.13. Waiting for the bootstrap process to complete	2435
18.2.14. Logging in to the cluster by using the CLI	2436
18.2.15. Approving the certificate signing requests for your machines	2437
18.2.16. Initial Operator configuration	2440
18.2.16.1. Image registry storage configuration	2441
18.2.16.1.1. Configuring registry storage for IBM Z	2441
18.2.16.1.2. Configuring storage for the image registry in non-production clusters	2442
18.2.17. Completing installation on user-provisioned infrastructure	2443
18.2.18. Telemetry access for OpenShift Container Platform	2446
18.2.19. Next steps	2446
18.3. INSTALLING A CLUSTER WITH Z/VM ON IBM Z AND IBM(R) LINUXONE IN A RESTRICTED NETWORK	2446
18.3.1. Prerequisites	2446
18.3.2. About installations in restricted networks	2447
18.3.2.1. Additional limits	2447
18.3.3. Internet access for OpenShift Container Platform	2448

18.3.4. Requirements for a cluster with user-provisioned infrastructure	2448
18.3.4.1. Required machines for cluster installation	2448
18.3.4.2. Minimum resource requirements for cluster installation	2449
18.3.4.3. Minimum IBM Z system environment	2449
Hardware requirements	2450
Operating system requirements	2450
IBM Z network connectivity requirements	2450
Disk storage for the z/VM guest virtual machines	2450
Storage / Main Memory	2450
18.3.4.4. Preferred IBM Z system environment	2451
Hardware requirements	2451
Operating system requirements	2451
IBM Z network connectivity requirements	2451
Disk storage for the z/VM guest virtual machines	2451
Storage / Main Memory	2451
18.3.4.5. Certificate signing requests management	2452
18.3.4.6. Networking requirements for user-provisioned infrastructure	2452
18.3.4.6.1. Setting the cluster node hostnames through DHCP	2452
18.3.4.6.2. Network connectivity requirements	2453
NTP configuration for user-provisioned infrastructure	2454
18.3.4.7. User-provisioned DNS requirements	2454
18.3.4.7.1. Example DNS configuration for user-provisioned clusters	2456
18.3.4.8. Load balancing requirements for user-provisioned infrastructure	2458
18.3.4.8.1. Example load balancer configuration for user-provisioned clusters	2460
18.3.5. Preparing the user-provisioned infrastructure	2462
18.3.6. Validating DNS resolution for user-provisioned infrastructure	2463
18.3.7. Generating a key pair for cluster node SSH access	2465
18.3.8. Manually creating the installation configuration file	2467
18.3.8.1. Installation configuration parameters	2468
18.3.8.1.1. Required configuration parameters	2468
18.3.8.1.2. Network configuration parameters	2469
18.3.8.1.3. Optional configuration parameters	2472
18.3.8.2. Sample install-config.yaml file for IBM Z	2477
18.3.8.3. Configuring the cluster-wide proxy during installation	2480
18.3.8.4. Configuring a three-node cluster	2482
18.3.9. Cluster Network Operator configuration	2483
18.3.9.1. Cluster Network Operator configuration object	2483
defaultNetwork object configuration	2484
Configuration for the OpenShift SDN network plugin	2485
Configuration for the OVN-Kubernetes network plugin	2486
kubeProxyConfig object configuration	2490
18.3.10. Creating the Kubernetes manifest and Ignition config files	2491
18.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2493
18.3.11.1. Advanced RHCOS installation reference	2496
18.3.11.1.1. Networking and bonding options for ISO installations	2496
Configuring DHCP or static IP addresses	2496
Configuring an IP address without a static hostname	2497
Specifying multiple network interfaces	2497
Configuring default gateway and route	2497
Disabling DHCP on a single interface	2498
Combining DHCP and static IP configurations	2498
Configuring VLANs on individual interfaces	2498
Providing multiple DNS servers	2498

Bonding multiple network interfaces to a single interface	2498
Bonding multiple network interfaces to a single interface	2499
Using network teaming	2499
18.3.12. Waiting for the bootstrap process to complete	2500
18.3.13. Logging in to the cluster by using the CLI	2500
18.3.14. Approving the certificate signing requests for your machines	2501
18.3.15. Initial Operator configuration	2504
18.3.15.1. Disabling the default OperatorHub catalog sources	2505
18.3.15.2. Image registry storage configuration	2505
18.3.15.2.1. Configuring registry storage for IBM Z	2505
18.3.15.2.2. Configuring storage for the image registry in non-production clusters	2507
18.3.16. Completing installation on user-provisioned infrastructure	2507
18.3.17. Next steps	2510
CHAPTER 19. INSTALLING WITH RHEL KVM ON IBM Z AND IBM LINUXONE	2511
19.1. PREPARING TO INSTALL WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE	2511
19.1.1. Prerequisites	2511
19.1.2. Choosing a method to install OpenShift Container Platform with RHEL KVM on IBM Z or IBM(R) LinuxONE	2511
19.2. INSTALLING A CLUSTER WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE	2511
19.2.1. Prerequisites	2511
19.2.2. Internet access for OpenShift Container Platform	2512
19.2.3. Machine requirements for a cluster with user-provisioned infrastructure	2512
19.2.3.1. Required machines	2512
19.2.3.2. Network connectivity requirements	2513
19.2.3.3. IBM Z network connectivity requirements	2513
19.2.3.4. Host machine resource requirements	2513
19.2.3.5. Minimum IBM Z system environment	2514
Hardware requirements	2514
Operating system requirements	2514
19.2.3.6. Minimum resource requirements	2514
19.2.3.7. Preferred IBM Z system environment	2515
Hardware requirements	2515
Operating system requirements	2515
19.2.3.8. Preferred resource requirements	2515
19.2.3.9. Certificate signing requests management	2516
19.2.3.10. Networking requirements for user-provisioned infrastructure	2516
19.2.3.10.1. Setting the cluster node hostnames through DHCP	2516
19.2.3.10.2. Network connectivity requirements	2517
NTP configuration for user-provisioned infrastructure	2518
19.2.3.11. User-provisioned DNS requirements	2518
19.2.3.11.1. Example DNS configuration for user-provisioned clusters	2520
19.2.3.12. Load balancing requirements for user-provisioned infrastructure	2522
19.2.3.12.1. Example load balancer configuration for user-provisioned clusters	2524
19.2.4. Preparing the user-provisioned infrastructure	2526
19.2.5. Validating DNS resolution for user-provisioned infrastructure	2528
19.2.6. Generating a key pair for cluster node SSH access	2530
19.2.7. Obtaining the installation program	2532
19.2.8. Installing the OpenShift CLI by downloading the binary	2533
Installing the OpenShift CLI on Linux	2533
Installing the OpenShift CLI on Windows	2533
Installing the OpenShift CLI on macOS	2534
19.2.9. Manually creating the installation configuration file	2534

19.2.9.1. Installation configuration parameters	2535
19.2.9.1.1. Required configuration parameters	2535
19.2.9.1.2. Network configuration parameters	2537
19.2.9.1.3. Optional configuration parameters	2539
19.2.9.2. Sample install-config.yaml file for IBM Z	2544
19.2.9.3. Configuring the cluster-wide proxy during installation	2547
19.2.9.4. Configuring a three-node cluster	2548
19.2.10. Cluster Network Operator configuration	2549
19.2.10.1. Cluster Network Operator configuration object	2550
defaultNetwork object configuration	2551
Configuration for the OpenShift SDN network plugin	2551
Configuration for the OVN-Kubernetes network plugin	2552
kubeProxyConfig object configuration	2556
19.2.11. Creating the Kubernetes manifest and Ignition config files	2557
19.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2559
19.2.12.1. Installing RHCOS using IBM Secure Execution	2559
19.2.12.2. Fast-track installation by using a prepackaged QCOW2 disk image	2561
19.2.12.3. Full installation on a new QCOW2 disk image	2562
19.2.12.4. Advanced RHCOS installation reference	2564
19.2.12.4.1. Networking options for ISO installations	2564
Configuring DHCP or static IP addresses	2564
Configuring an IP address without a static hostname	2565
Specifying multiple network interfaces	2565
Configuring default gateway and route	2565
Disabling DHCP on a single interface	2565
Combining DHCP and static IP configurations	2566
Configuring VLANs on individual interfaces	2566
Providing multiple DNS servers	2566
19.2.13. Waiting for the bootstrap process to complete	2566
19.2.14. Logging in to the cluster by using the CLI	2567
19.2.15. Approving the certificate signing requests for your machines	2568
19.2.16. Initial Operator configuration	2570
19.2.16.1. Image registry storage configuration	2571
19.2.16.1.1. Configuring registry storage for IBM Z	2571
19.2.16.1.2. Configuring storage for the image registry in non-production clusters	2573
19.2.17. Completing installation on user-provisioned infrastructure	2574
19.2.18. Telemetry access for OpenShift Container Platform	2576
19.2.19. Next steps	2576
19.3. INSTALLING A CLUSTER WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE IN A RESTRICTED NETWORK	2576
19.3.1. Prerequisites	2577
19.3.2. About installations in restricted networks	2577
19.3.2.1. Additional limits	2578
19.3.3. Internet access for OpenShift Container Platform	2578
19.3.4. Machine requirements for a cluster with user-provisioned infrastructure	2578
19.3.4.1. Required machines	2578
19.3.4.2. Network connectivity requirements	2579
19.3.4.3. IBM Z network connectivity requirements	2579
19.3.4.4. Host machine resource requirements	2579
19.3.4.5. Minimum IBM Z system environment	2580
Hardware requirements	2580
Operating system requirements	2580
19.3.4.6. Minimum resource requirements	2580

19.3.4.7. Preferred IBM Z system environment	2581
Hardware requirements	2581
Operating system requirements	2581
19.3.4.8. Preferred resource requirements	2581
19.3.4.9. Certificate signing requests management	2582
19.3.4.10. Networking requirements for user-provisioned infrastructure	2582
19.3.4.10.1. Setting the cluster node hostnames through DHCP	2582
19.3.4.10.2. Network connectivity requirements	2583
NTP configuration for user-provisioned infrastructure	2584
19.3.4.11. User-provisioned DNS requirements	2584
19.3.4.11.1. Example DNS configuration for user-provisioned clusters	2586
19.3.4.12. Load balancing requirements for user-provisioned infrastructure	2588
19.3.4.12.1. Example load balancer configuration for user-provisioned clusters	2590
19.3.5. Preparing the user-provisioned infrastructure	2592
19.3.6. Validating DNS resolution for user-provisioned infrastructure	2594
19.3.7. Generating a key pair for cluster node SSH access	2596
19.3.8. Manually creating the installation configuration file	2598
19.3.8.1. Installation configuration parameters	2598
19.3.8.1.1. Required configuration parameters	2599
19.3.8.1.2. Network configuration parameters	2600
19.3.8.1.3. Optional configuration parameters	2602
19.3.8.2. Sample install-config.yaml file for IBM Z	2608
19.3.8.3. Configuring the cluster-wide proxy during installation	2611
19.3.8.4. Configuring a three-node cluster	2613
19.3.9. Cluster Network Operator configuration	2614
19.3.9.1. Cluster Network Operator configuration object	2615
defaultNetwork object configuration	2615
Configuration for the OpenShift SDN network plugin	2616
Configuration for the OVN-Kubernetes network plugin	2617
kubeProxyConfig object configuration	2621
19.3.10. Creating the Kubernetes manifest and Ignition config files	2622
19.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2624
19.3.11.1. Installing RHCOS using IBM Secure Execution	2624
19.3.11.2. Fast-track installation by using a prepackaged QCOW2 disk image	2626
19.3.11.3. Full installation on a new QCOW2 disk image	2627
19.3.11.4. Advanced RHCOS installation reference	2629
19.3.11.4.1. Networking options for ISO installations	2629
Configuring DHCP or static IP addresses	2629
Configuring an IP address without a static hostname	2630
Specifying multiple network interfaces	2630
Configuring default gateway and route	2630
Disabling DHCP on a single interface	2630
Combining DHCP and static IP configurations	2631
Configuring VLANs on individual interfaces	2631
Providing multiple DNS servers	2631
19.3.12. Waiting for the bootstrap process to complete	2631
19.3.13. Logging in to the cluster by using the CLI	2632
19.3.14. Approving the certificate signing requests for your machines	2633
19.3.15. Initial Operator configuration	2635
19.3.15.1. Disabling the default OperatorHub catalog sources	2636
19.3.15.2. Image registry storage configuration	2637
19.3.15.2.1. Configuring registry storage for IBM Z	2637
19.3.15.2.2. Configuring storage for the image registry in non-production clusters	2639

19.3.16. Completing installation on user-provisioned infrastructure	2639
19.3.17. Next steps	2642
CHAPTER 20. INSTALLING ON IBM POWER	2643
20.1. PREPARING TO INSTALL ON IBM POWER	2643
20.1.1. Prerequisites	2643
20.1.2. Choosing a method to install OpenShift Container Platform on IBM Power	2643
20.2. INSTALLING A CLUSTER ON IBM POWER	2643
20.2.1. Prerequisites	2643
20.2.2. Internet access for OpenShift Container Platform	2644
20.2.3. Requirements for a cluster with user-provisioned infrastructure	2644
20.2.3.1. Required machines for cluster installation	2644
20.2.3.2. Minimum resource requirements for cluster installation	2645
20.2.3.3. Minimum IBM Power requirements	2645
Hardware requirements	2646
Operating system requirements	2646
Disk storage for the IBM Power guest virtual machines	2646
Network for the PowerVM guest virtual machines	2646
Storage / main memory	2646
20.2.3.4. Recommended IBM Power system requirements	2646
Hardware requirements	2646
Operating system requirements	2646
Disk storage for the IBM Power guest virtual machines	2647
Network for the PowerVM guest virtual machines	2647
Storage / main memory	2647
20.2.3.5. Certificate signing requests management	2647
20.2.3.6. Networking requirements for user-provisioned infrastructure	2647
20.2.3.6.1. Setting the cluster node hostnames through DHCP	2648
20.2.3.6.2. Network connectivity requirements	2648
NTP configuration for user-provisioned infrastructure	2649
20.2.3.7. User-provisioned DNS requirements	2650
20.2.3.7.1. Example DNS configuration for user-provisioned clusters	2651
20.2.3.8. Load balancing requirements for user-provisioned infrastructure	2654
20.2.3.8.1. Example load balancer configuration for user-provisioned clusters	2655
20.2.4. Preparing the user-provisioned infrastructure	2657
20.2.5. Validating DNS resolution for user-provisioned infrastructure	2659
20.2.6. Generating a key pair for cluster node SSH access	2661
20.2.7. Obtaining the installation program	2663
20.2.8. Installing the OpenShift CLI by downloading the binary	2664
Installing the OpenShift CLI on Linux	2664
Installing the OpenShift CLI on Windows	2665
Installing the OpenShift CLI on macOS	2665
20.2.9. Manually creating the installation configuration file	2666
20.2.9.1. Installation configuration parameters	2667
20.2.9.1.1. Required configuration parameters	2667
20.2.9.1.2. Network configuration parameters	2668
20.2.9.1.3. Optional configuration parameters	2670
20.2.9.2. Sample install-config.yaml file for IBM Power	2676
20.2.9.3. Configuring the cluster-wide proxy during installation	2679
20.2.9.4. Configuring a three-node cluster	2681
20.2.10. Cluster Network Operator configuration	2682
20.2.10.1. Cluster Network Operator configuration object	2682
defaultNetwork object configuration	2683

Configuration for the OpenShift SDN network plugin	2683
Configuration for the OVN-Kubernetes network plugin	2684
kubeProxyConfig object configuration	2688
20.2.11. Creating the Kubernetes manifest and Ignition config files	2689
20.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2691
20.2.12.1. Installing RHCOS by using an ISO image	2691
20.2.12.1.1. Advanced RHCOS installation reference	2694
20.2.12.1.1.1. Networking and bonding options for ISO installations	2694
Configuring DHCP or static IP addresses	2695
Configuring an IP address without a static hostname	2695
Specifying multiple network interfaces	2696
Configuring default gateway and route	2696
Disabling DHCP on a single interface	2696
Combining DHCP and static IP configurations	2696
Configuring VLANs on individual interfaces	2696
Providing multiple DNS servers	2697
Bonding multiple network interfaces to a single interface	2697
Bonding multiple network interfaces to a single interface	2697
Using network teaming	2698
20.2.12.2. Installing RHCOS by using PXE booting	2698
20.2.12.3. Enabling multipathing with kernel arguments on RHCOS	2702
20.2.13. Waiting for the bootstrap process to complete	2703
20.2.14. Logging in to the cluster by using the CLI	2704
20.2.15. Approving the certificate signing requests for your machines	2705
20.2.16. Initial Operator configuration	2707
20.2.16.1. Image registry storage configuration	2708
20.2.16.1.1. Configuring registry storage for IBM Power	2709
20.2.16.1.2. Configuring storage for the image registry in non-production clusters	2710
20.2.17. Completing installation on user-provisioned infrastructure	2711
20.2.18. Telemetry access for OpenShift Container Platform	2713
20.2.19. Next steps	2713
20.3. INSTALLING A CLUSTER ON IBM POWER IN A RESTRICTED NETWORK	2713
20.3.1. Prerequisites	2714
20.3.2. About installations in restricted networks	2714
20.3.2.1. Additional limits	2715
20.3.3. Internet access for OpenShift Container Platform	2715
20.3.4. Requirements for a cluster with user-provisioned infrastructure	2715
20.3.4.1. Required machines for cluster installation	2715
20.3.4.2. Minimum resource requirements for cluster installation	2716
20.3.4.3. Minimum IBM Power requirements	2717
Hardware requirements	2717
Operating system requirements	2717
Disk storage for the IBM Power guest virtual machines	2717
Network for the PowerVM guest virtual machines	2717
Storage / main memory	2717
20.3.4.4. Recommended IBM Power system requirements	2718
Hardware requirements	2718
Operating system requirements	2718
Disk storage for the IBM Power guest virtual machines	2718
Network for the PowerVM guest virtual machines	2718
Storage / main memory	2718
20.3.4.5. Certificate signing requests management	2718
20.3.4.6. Networking requirements for user-provisioned infrastructure	2718

20.3.4.6.1. Setting the cluster node hostnames through DHCP	2719
20.3.4.6.2. Network connectivity requirements	2719
NTP configuration for user-provisioned infrastructure	2720
20.3.4.7. User-provisioned DNS requirements	2721
20.3.4.7.1. Example DNS configuration for user-provisioned clusters	2723
20.3.4.8. Load balancing requirements for user-provisioned infrastructure	2725
20.3.4.8.1. Example load balancer configuration for user-provisioned clusters	2727
20.3.5. Preparing the user-provisioned infrastructure	2728
20.3.6. Validating DNS resolution for user-provisioned infrastructure	2730
20.3.7. Generating a key pair for cluster node SSH access	2733
20.3.8. Manually creating the installation configuration file	2734
20.3.8.1. Installation configuration parameters	2735
20.3.8.1.1. Required configuration parameters	2735
20.3.8.1.2. Network configuration parameters	2737
20.3.8.1.3. Optional configuration parameters	2739
20.3.8.2. Sample install-config.yaml file for IBM Power	2744
20.3.8.3. Configuring the cluster-wide proxy during installation	2747
20.3.8.4. Configuring a three-node cluster	2749
20.3.9. Cluster Network Operator configuration	2750
20.3.9.1. Cluster Network Operator configuration object	2750
defaultNetwork object configuration	2751
Configuration for the OpenShift SDN network plugin	2751
Configuration for the OVN-Kubernetes network plugin	2752
kubeProxyConfig object configuration	2756
20.3.10. Creating the Kubernetes manifest and Ignition config files	2757
20.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	2759
20.3.11.1. Installing RHCOS by using an ISO image	2759
20.3.11.1.1. Advanced RHCOS installation reference	2762
20.3.11.1.1.1. Networking and bonding options for ISO installations	2762
Configuring DHCP or static IP addresses	2763
Configuring an IP address without a static hostname	2763
Specifying multiple network interfaces	2764
Configuring default gateway and route	2764
Disabling DHCP on a single interface	2764
Combining DHCP and static IP configurations	2764
Configuring VLANs on individual interfaces	2764
Providing multiple DNS servers	2765
Bonding multiple network interfaces to a single interface	2765
Bonding multiple network interfaces to a single interface	2765
Using network teaming	2766
20.3.11.2. Installing RHCOS by using PXE booting	2766
20.3.11.3. Enabling multipathing with kernel arguments on RHCOS	2770
20.3.12. Waiting for the bootstrap process to complete	2771
20.3.13. Logging in to the cluster by using the CLI	2772
20.3.14. Approving the certificate signing requests for your machines	2773
20.3.15. Initial Operator configuration	2775
20.3.15.1. Disabling the default OperatorHub catalog sources	2776
20.3.15.2. Image registry storage configuration	2777
20.3.15.2.1. Changing the image registry's management state	2777
20.3.15.2.2. Configuring registry storage for IBM Power	2777
20.3.15.2.3. Configuring storage for the image registry in non-production clusters	2779
20.3.16. Completing installation on user-provisioned infrastructure	2779
20.3.17. Next steps	2782

CHAPTER 21. INSTALLING ON OPENSTACK	2783
21.1. PREPARING TO INSTALL ON OPENSTACK	2783
21.1.1. Prerequisites	2783
21.1.2. Choosing a method to install OpenShift Container Platform on OpenStack	2783
21.1.2.1. Installing a cluster on installer-provisioned infrastructure	2783
21.1.2.2. Installing a cluster on user-provisioned infrastructure	2783
21.1.3. Scanning RHOSP endpoints for legacy HTTPS certificates	2784
21.1.3.1. Scanning RHOSP endpoints for legacy HTTPS certificates manually	2786
21.2. PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV OR OVS-DPDK ON OPENSTACK	2787
21.2.1. Requirements for clusters on RHOSP that use either SR-IOV or OVS-DPDK	2787
21.2.1.1. Requirements for clusters on RHOSP that use SR-IOV	2787
21.2.1.2. Requirements for clusters on RHOSP that use OVS-DPDK	2788
21.2.2. Preparing to install a cluster that uses SR-IOV	2788
21.2.2.1. Creating SR-IOV networks for compute machines	2788
21.2.3. Preparing to install a cluster that uses OVS-DPDK	2789
21.2.4. Next steps	2789
21.3. INSTALLING A CLUSTER ON OPENSTACK WITH CUSTOMIZATIONS	2790
21.3.1. Prerequisites	2790
21.3.2. Resource guidelines for installing OpenShift Container Platform on RHOSP	2790
21.3.2.1. Control plane machines	2791
21.3.2.2. Compute machines	2791
21.3.2.3. Bootstrap machine	2792
21.3.3. Internet access for OpenShift Container Platform	2792
21.3.4. Enabling Swift on RHOSP	2792
21.3.5. Configuring an image registry with custom storage on clusters that run on RHOSP	2793
21.3.6. Verifying external network access	2795
21.3.7. Defining parameters for the installation program	2796
21.3.8. Setting OpenStack Cloud Controller Manager options	2798
21.3.9. Obtaining the installation program	2800
21.3.10. Creating the installation configuration file	2801
21.3.10.1. Configuring the cluster-wide proxy during installation	2802
21.3.11. Installation configuration parameters	2804
21.3.11.1. Required configuration parameters	2804
21.3.11.2. Network configuration parameters	2805
21.3.11.3. Optional configuration parameters	2807
21.3.11.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters	2812
21.3.11.5. Optional RHOSP configuration parameters	2813
21.3.11.6. Custom subnets in RHOSP deployments	2818
21.3.11.7. Deploying a cluster with bare metal machines	2819
21.3.11.8. Cluster deployment on RHOSP provider networks	2821
21.3.11.8.1. RHOSP provider network requirements for cluster installation	2822
21.3.11.8.2. Deploying a cluster that has a primary interface on a provider network	2823
21.3.11.9. Sample customized install-config.yaml file for RHOSP	2824
21.3.12. Generating a key pair for cluster node SSH access	2825
21.3.13. Enabling access to the environment	2827
21.3.13.1. Enabling access with floating IP addresses	2827
21.3.13.2. Completing installation without floating IP addresses	2828
21.3.14. Deploying the cluster	2829
21.3.15. Verifying cluster status	2830
21.3.16. Logging in to the cluster by using the CLI	2831
21.3.17. Telemetry access for OpenShift Container Platform	2832
21.3.18. Next steps	2832
21.4. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR	2832

21.4.1. Prerequisites	2832
21.4.2. About Kuryr SDN	2833
21.4.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr	2834
21.4.3.1. Increasing quota	2836
21.4.3.2. Configuring Neutron	2836
21.4.3.3. Configuring Octavia	2836
21.4.3.3.1. The Octavia OVN Driver	2838
21.4.3.4. Known limitations of installing with Kuryr	2838
RHOSP general limitations	2838
RHOSP version limitations	2839
RHOSP upgrade limitations	2839
21.4.3.5. Control plane machines	2839
21.4.3.6. Compute machines	2840
21.4.3.7. Bootstrap machine	2840
21.4.4. Internet access for OpenShift Container Platform	2840
21.4.5. Enabling Swift on RHOSP	2841
21.4.6. Verifying external network access	2841
21.4.7. Defining parameters for the installation program	2842
21.4.8. Setting OpenStack Cloud Controller Manager options	2844
21.4.9. Obtaining the installation program	2846
21.4.10. Creating the installation configuration file	2847
21.4.10.1. Configuring the cluster-wide proxy during installation	2848
21.4.11. Installation configuration parameters	2850
21.4.11.1. Required configuration parameters	2850
21.4.11.2. Network configuration parameters	2851
21.4.11.3. Optional configuration parameters	2853
21.4.11.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters	2858
21.4.11.5. Optional RHOSP configuration parameters	2859
21.4.11.6. Custom subnets in RHOSP deployments	2864
21.4.11.7. Sample customized install-config.yaml file for RHOSP with Kuryr	2865
21.4.11.8. Cluster deployment on RHOSP provider networks	2866
21.4.11.8.1. RHOSP provider network requirements for cluster installation	2867
21.4.11.8.2. Deploying a cluster that has a primary interface on a provider network	2868
21.4.11.9. Kuryr ports pools	2870
21.4.11.10. Adjusting Kuryr ports pools during installation	2870
21.4.12. Generating a key pair for cluster node SSH access	2872
21.4.13. Enabling access to the environment	2874
21.4.13.1. Enabling access with floating IP addresses	2874
21.4.13.2. Completing installation without floating IP addresses	2875
21.4.14. Deploying the cluster	2876
21.4.15. Verifying cluster status	2877
21.4.16. Logging in to the cluster by using the CLI	2878
21.4.17. Telemetry access for OpenShift Container Platform	2879
21.4.18. Next steps	2879
21.5. INSTALLING A CLUSTER ON OPENSTACK ON YOUR OWN INFRASTRUCTURE	2879
21.5.1. Prerequisites	2879
21.5.2. Internet access for OpenShift Container Platform	2880
21.5.3. Resource guidelines for installing OpenShift Container Platform on RHOSP	2880
21.5.3.1. Control plane machines	2881
21.5.3.2. Compute machines	2882
21.5.3.3. Bootstrap machine	2882
21.5.4. Downloading playbook dependencies	2882
21.5.5. Downloading the installation playbooks	2883

21.5.6. Obtaining the installation program	2884
21.5.7. Generating a key pair for cluster node SSH access	2885
21.5.8. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image	2887
21.5.9. Verifying external network access	2888
21.5.10. Enabling access to the environment	2889
21.5.10.1. Enabling access with floating IP addresses	2889
21.5.10.2. Completing installation without floating IP addresses	2890
21.5.11. Defining parameters for the installation program	2891
21.5.12. Creating the installation configuration file	2892
21.5.13. Installation configuration parameters	2894
21.5.13.1. Required configuration parameters	2894
21.5.13.2. Network configuration parameters	2895
21.5.13.3. Optional configuration parameters	2897
21.5.13.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters	2902
21.5.13.5. Optional RHOSP configuration parameters	2903
21.5.13.6. Custom subnets in RHOSP deployments	2908
21.5.13.7. Sample customized install-config.yaml file for RHOSP	2909
21.5.13.8. Setting a custom subnet for machines	2910
21.5.13.9. Emptying compute machine pools	2910
21.5.13.10. Cluster deployment on RHOSP provider networks	2911
21.5.13.10.1. RHOSP provider network requirements for cluster installation	2912
21.5.13.10.2. Deploying a cluster that has a primary interface on a provider network	2913
21.5.14. Creating the Kubernetes manifest and Ignition config files	2915
21.5.15. Preparing the bootstrap Ignition files	2916
21.5.16. Creating control plane Ignition config files on RHOSP	2919
21.5.17. Creating network resources on RHOSP	2920
21.5.17.1. Deploying a cluster with bare metal machines	2921
21.5.18. Creating the bootstrap machine on RHOSP	2923
21.5.19. Creating the control plane machines on RHOSP	2923
21.5.20. Logging in to the cluster by using the CLI	2924
21.5.21. Deleting bootstrap resources from RHOSP	2925
21.5.22. Creating compute machines on RHOSP	2925
21.5.23. Approving the certificate signing requests for your machines	2926
21.5.24. Verifying a successful installation	2929
21.5.25. Telemetry access for OpenShift Container Platform	2929
21.5.26. Next steps	2929
21.6. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR ON YOUR OWN INFRASTRUCTURE	2930
21.6.1. Prerequisites	2930
21.6.2. About Kuryr SDN	2930
21.6.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr	2932
21.6.3.1. Increasing quota	2933
21.6.3.2. Configuring Neutron	2934
21.6.3.3. Configuring Octavia	2934
21.6.3.3.1. The Octavia OVN Driver	2935
21.6.3.4. Known limitations of installing with Kuryr	2936
RHOSP general limitations	2936
RHOSP version limitations	2936
RHOSP upgrade limitations	2937
21.6.3.5. Control plane machines	2937
21.6.3.6. Compute machines	2937
21.6.3.7. Bootstrap machine	2937
21.6.4. Internet access for OpenShift Container Platform	2938
21.6.5. Downloading playbook dependencies	2938

21.6.6. Downloading the installation playbooks	2939
21.6.7. Obtaining the installation program	2940
21.6.8. Generating a key pair for cluster node SSH access	2941
21.6.9. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image	2943
21.6.10. Verifying external network access	2944
21.6.11. Enabling access to the environment	2945
21.6.11.1. Enabling access with floating IP addresses	2945
21.6.11.2. Completing installation without floating IP addresses	2946
21.6.12. Defining parameters for the installation program	2947
21.6.13. Creating the installation configuration file	2948
21.6.14. Installation configuration parameters	2950
21.6.14.1. Required configuration parameters	2950
21.6.14.2. Network configuration parameters	2951
21.6.14.3. Optional configuration parameters	2953
21.6.14.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters	2958
21.6.14.5. Optional RHOSP configuration parameters	2959
21.6.14.6. Custom subnets in RHOSP deployments	2964
21.6.14.7. Sample customized install-config.yaml file for RHOSP with Kuryr	2965
21.6.14.8. Cluster deployment on RHOSP provider networks	2966
21.6.14.8.1. RHOSP provider network requirements for cluster installation	2967
21.6.14.8.2. Deploying a cluster that has a primary interface on a provider network	2968
21.6.14.9. Kuryr ports pools	2970
21.6.14.10. Adjusting Kuryr ports pools during installation	2970
21.6.14.11. Setting a custom subnet for machines	2972
21.6.14.12. Emptying compute machine pools	2973
21.6.14.13. Modifying the network type	2973
21.6.15. Creating the Kubernetes manifest and Ignition config files	2974
21.6.16. Preparing the bootstrap Ignition files	2975
21.6.17. Creating control plane Ignition config files on RHOSP	2978
21.6.18. Creating network resources on RHOSP	2979
21.6.19. Creating the bootstrap machine on RHOSP	2980
21.6.20. Creating the control plane machines on RHOSP	2981
21.6.21. Logging in to the cluster by using the CLI	2982
21.6.22. Deleting bootstrap resources from RHOSP	2982
21.6.23. Creating compute machines on RHOSP	2983
21.6.24. Approving the certificate signing requests for your machines	2984
21.6.25. Verifying a successful installation	2986
21.6.26. Telemetry access for OpenShift Container Platform	2987
21.6.27. Next steps	2987
21.7. INSTALLING A CLUSTER ON OPENSTACK IN A RESTRICTED NETWORK	2987
21.7.1. Prerequisites	2987
21.7.2. About installations in restricted networks	2988
21.7.2.1. Additional limits	2988
21.7.3. Resource guidelines for installing OpenShift Container Platform on RHOSP	2988
21.7.3.1. Control plane machines	2989
21.7.3.2. Compute machines	2989
21.7.3.3. Bootstrap machine	2990
21.7.4. Internet access for OpenShift Container Platform	2990
21.7.5. Enabling Swift on RHOSP	2990
21.7.6. Defining parameters for the installation program	2991
21.7.7. Setting OpenStack Cloud Controller Manager options	2993
21.7.8. Creating the RHCOS image for restricted network installations	2994
21.7.9. Creating the installation configuration file	2995

21.7.9.1. Configuring the cluster-wide proxy during installation	2998
21.7.9.2. Installation configuration parameters	3000
21.7.9.2.1. Required configuration parameters	3000
21.7.9.2.2. Network configuration parameters	3001
21.7.9.2.3. Optional configuration parameters	3003
21.7.9.2.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters	3008
21.7.9.2.5. Optional RHOSP configuration parameters	3009
21.7.9.3. Sample customized install-config.yaml file for restricted OpenStack installations	3014
21.7.10. Generating a key pair for cluster node SSH access	3015
21.7.11. Enabling access to the environment	3017
21.7.11.1. Enabling access with floating IP addresses	3017
21.7.11.2. Completing installation without floating IP addresses	3018
21.7.12. Deploying the cluster	3019
21.7.13. Verifying cluster status	3020
21.7.14. Logging in to the cluster by using the CLI	3021
21.7.15. Disabling the default OperatorHub catalog sources	3022
21.7.16. Telemetry access for OpenShift Container Platform	3022
21.7.17. Next steps	3022
21.8. OPENSTACK CLOUD CONTROLLER MANAGER REFERENCE GUIDE	3023
21.8.1. The OpenStack Cloud Controller Manager	3023
21.8.2. The OpenStack Cloud Controller Manager (CCM) config map	3024
21.8.2.1. Load balancer options	3024
21.8.2.2. Options that the Operator overrides	3027
21.9. UNINSTALLING A CLUSTER ON OPENSTACK	3028
21.9.1. Removing a cluster that uses installer-provisioned infrastructure	3029
21.10. UNINSTALLING A CLUSTER ON RHOSP FROM YOUR OWN INFRASTRUCTURE	3029
21.10.1. Downloading playbook dependencies	3029
21.10.2. Removing a cluster from RHOSP that uses your own infrastructure	3030
CHAPTER 22. INSTALLING ON RHV	3032
22.1. PREPARING TO INSTALL ON RED HAT VIRTUALIZATION (RHV)	3032
22.1.1. Prerequisites	3032
22.1.2. Choosing a method to install OpenShift Container Platform on RHV	3032
22.1.2.1. Installing a cluster on installer-provisioned infrastructure	3032
22.1.2.2. Installing a cluster on user-provisioned infrastructure	3032
22.2. INSTALLING A CLUSTER QUICKLY ON RHV	3033
22.2.1. Prerequisites	3033
22.2.2. Internet access for OpenShift Container Platform	3034
22.2.3. Requirements for the RHV environment	3034
22.2.4. Verifying the requirements for the RHV environment	3036
22.2.5. Preparing the network environment on RHV	3038
22.2.6. Installing OpenShift Container Platform on RHV in insecure mode	3038
22.2.7. Generating a key pair for cluster node SSH access	3039
22.2.8. Obtaining the installation program	3041
22.2.9. Deploying the cluster	3042
22.2.10. Installing the OpenShift CLI by downloading the binary	3044
Installing the OpenShift CLI on Linux	3045
Installing the OpenShift CLI on Windows	3045
Installing the OpenShift CLI on macOS	3046
22.2.11. Logging in to the cluster by using the CLI	3046
22.2.12. Verifying cluster status	3047
22.2.13. Accessing the OpenShift Container Platform web console on RHV	3048
22.2.14. Telemetry access for OpenShift Container Platform	3048

22.2.15. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)	3048
22.2.15.1. CPU load increases and nodes go into a Not Ready state	3049
22.2.15.2. Trouble connecting the OpenShift Container Platform cluster API	3049
22.2.16. Post-installation tasks	3049
22.3. INSTALLING A CLUSTER ON RHV WITH CUSTOMIZATIONS	3050
22.3.1. Prerequisites	3050
22.3.2. Internet access for OpenShift Container Platform	3051
22.3.3. Requirements for the RHV environment	3051
22.3.4. Verifying the requirements for the RHV environment	3053
22.3.5. Preparing the network environment on RHV	3055
22.3.6. Installing OpenShift Container Platform on RHV in insecure mode	3055
22.3.7. Generating a key pair for cluster node SSH access	3056
22.3.8. Obtaining the installation program	3058
22.3.9. Creating the installation configuration file	3059
22.3.9.1. Example install-config.yaml files for Red Hat Virtualization (RHV)	3062
Example default install-config.yaml file	3062
Example minimal install-config.yaml file	3063
Example Custom machine pools in an install-config.yaml file	3064
Example non-enforcing affinity group	3065
Example removing all affinity groups for a non-production lab setup	3065
22.3.9.2. Installation configuration parameters	3066
22.3.9.2.1. Required configuration parameters	3066
22.3.9.2.2. Network configuration parameters	3067
22.3.9.2.3. Optional configuration parameters	3069
22.3.9.2.4. Additional Red Hat Virtualization (RHV) configuration parameters	3075
22.3.9.2.5. Additional RHV parameters for machine pools	3078
22.3.10. Deploying the cluster	3081
22.3.11. Installing the OpenShift CLI by downloading the binary	3082
Installing the OpenShift CLI on Linux	3082
Installing the OpenShift CLI on Windows	3083
Installing the OpenShift CLI on macOS	3083
22.3.12. Logging in to the cluster by using the CLI	3084
22.3.13. Verifying cluster status	3085
22.3.14. Accessing the OpenShift Container Platform web console on RHV	3085
22.3.15. Telemetry access for OpenShift Container Platform	3086
22.3.16. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)	3086
22.3.16.1. CPU load increases and nodes go into a Not Ready state	3086
22.3.16.2. Trouble connecting the OpenShift Container Platform cluster API	3087
22.3.17. Post-installation tasks	3087
22.3.18. Next steps	3087
22.4. INSTALLING A CLUSTER ON RHV WITH USER-PROVISIONED INFRASTRUCTURE	3087
22.4.1. Prerequisites	3088
22.4.2. Internet access for OpenShift Container Platform	3088
22.4.3. Requirements for the RHV environment	3089
22.4.4. Verifying the requirements for the RHV environment	3090
22.4.5. Networking requirements for user-provisioned infrastructure	3092
22.4.5.1. Setting the cluster node hostnames through DHCP	3093
22.4.5.2. Network connectivity requirements	3093
NTP configuration for user-provisioned infrastructure	3095
22.4.6. Setting up the installation machine	3095
22.4.7. Installing OpenShift Container Platform on RHV in insecure mode	3095
22.4.8. Generating a key pair for cluster node SSH access	3096
22.4.9. Obtaining the installation program	3098

22.4.10. Downloading the Ansible playbooks	3099
22.4.11. The inventory.yml file	3099
22.4.12. Specifying the RHCOS image settings	3103
22.4.13. Creating the install config file	3104
22.4.14. Customizing install-config.yaml	3105
22.4.15. Generate manifest files	3106
22.4.16. Making control-plane nodes non-schedulable	3108
22.4.17. Building the Ignition files	3108
22.4.18. Creating templates and virtual machines	3109
22.4.19. Creating the bootstrap machine	3110
22.4.20. Creating the control plane nodes	3110
22.4.21. Verifying cluster status	3111
22.4.22. Removing the bootstrap machine	3111
22.4.23. Creating the worker nodes and completing the installation	3112
22.4.24. Telemetry access for OpenShift Container Platform	3113
22.5. INSTALLING A CLUSTER ON RHV IN A RESTRICTED NETWORK	3114
22.5.1. Prerequisites	3114
22.5.2. About installations in restricted networks	3114
22.5.2.1. Additional limits	3115
22.5.3. Internet access for OpenShift Container Platform	3115
22.5.4. Requirements for the RHV environment	3115
22.5.5. Verifying the requirements for the RHV environment	3117
22.5.6. Networking requirements for user-provisioned infrastructure	3119
22.5.6.1. Setting the cluster node hostnames through DHCP	3119
22.5.6.2. Network connectivity requirements	3120
NTP configuration for user-provisioned infrastructure	3121
22.5.7. User-provisioned DNS requirements	3121
22.5.7.1. Example DNS configuration for user-provisioned clusters	3123
22.5.7.2. Load balancing requirements for user-provisioned infrastructure	3125
22.5.7.2.1. Example load balancer configuration for user-provisioned clusters	3127
22.5.8. Setting up the installation machine	3129
22.5.9. Setting up the CA certificate for RHV	3130
22.5.10. Generating a key pair for cluster node SSH access	3131
22.5.11. Downloading the Ansible playbooks	3132
22.5.12. The inventory.yml file	3133
22.5.13. Specifying the RHCOS image settings	3137
22.5.14. Creating the install config file	3137
22.5.15. Sample install-config.yaml file for RHV	3139
22.5.15.1. Configuring the cluster-wide proxy during installation	3141
22.5.16. Customizing install-config.yaml	3143
22.5.17. Generate manifest files	3144
22.5.18. Making control-plane nodes non-schedulable	3145
22.5.19. Building the Ignition files	3146
22.5.20. Creating templates and virtual machines	3147
22.5.21. Creating the bootstrap machine	3147
22.5.22. Creating the control plane nodes	3148
22.5.23. Verifying cluster status	3149
22.5.24. Removing the bootstrap machine	3149
22.5.25. Creating the worker nodes and completing the installation	3149
22.5.26. Telemetry access for OpenShift Container Platform	3151
22.5.27. Disabling the default OperatorHub catalog sources	3151
22.6. UNINSTALLING A CLUSTER ON RHV	3152
22.6.1. Removing a cluster that uses installer-provisioned infrastructure	3152

22.6.2. Removing a cluster that uses user-provisioned infrastructure	3153
CHAPTER 23. INSTALLING ON VSPHERE	3154
23.1. PREPARING TO INSTALL ON VSPHERE	3154
23.1.1. Prerequisites	3154
23.1.2. Choosing a method to install OpenShift Container Platform on vSphere	3154
23.1.2.1. Installer-provisioned infrastructure installation of OpenShift Container Platform on vSphere	3154
23.1.2.2. User-provisioned infrastructure installation of OpenShift Container Platform on vSphere	3155
23.1.3. VMware vSphere infrastructure requirements	3155
23.1.4. VMware vSphere CSI Driver Operator requirements	3156
23.1.5. Configuring the vSphere connection settings	3157
23.1.6. Uninstalling an installer-provisioned infrastructure installation of OpenShift Container Platform on vSphere	3157
23.2. INSTALLING A CLUSTER ON VSPHERE	3157
23.2.1. Prerequisites	3157
23.2.2. Internet access for OpenShift Container Platform	3158
23.2.3. VMware vSphere infrastructure requirements	3158
23.2.4. Network connectivity requirements	3160
23.2.5. VMware vSphere CSI Driver Operator requirements	3161
23.2.6. vCenter requirements	3161
Required vCenter account privileges	3161
Using OpenShift Container Platform with vMotion	3170
Cluster resources	3170
Cluster limits	3171
Networking requirements	3171
Required IP Addresses	3171
DNS records	3171
23.2.7. Generating a key pair for cluster node SSH access	3172
23.2.8. Obtaining the installation program	3174
23.2.9. Adding vCenter root CA certificates to your system trust	3175
23.2.10. Deploying the cluster	3176
23.2.11. Installing the OpenShift CLI by downloading the binary	3178
Installing the OpenShift CLI on Linux	3178
Installing the OpenShift CLI on Windows	3179
Installing the OpenShift CLI on macOS	3179
23.2.12. Logging in to the cluster by using the CLI	3180
23.2.13. Creating registry storage	3181
23.2.13.1. Image registry removed during installation	3181
23.2.13.2. Image registry storage configuration	3181
23.2.13.2.1. Configuring registry storage for VMware vSphere	3181
23.2.13.2.2. Configuring block registry storage for VMware vSphere	3183
23.2.14. Backing up VMware vSphere volumes	3184
23.2.15. Telemetry access for OpenShift Container Platform	3184
23.2.16. Next steps	3185
23.3. INSTALLING A CLUSTER ON VSPHERE WITH CUSTOMIZATIONS	3185
23.3.1. Prerequisites	3185
23.3.2. Internet access for OpenShift Container Platform	3186
23.3.3. VMware vSphere infrastructure requirements	3186
23.3.4. Network connectivity requirements	3187
23.3.5. VMware vSphere CSI Driver Operator requirements	3188
23.3.6. vCenter requirements	3189
Required vCenter account privileges	3189
Using OpenShift Container Platform with vMotion	3198

Cluster resources	3198
Cluster limits	3199
Networking requirements	3199
Required IP Addresses	3199
DNS records	3199
23.3.7. Generating a key pair for cluster node SSH access	3200
23.3.8. Obtaining the installation program	3202
23.3.9. Adding vCenter root CA certificates to your system trust	3203
23.3.10. VMware vSphere region and zone enablement	3203
23.3.11. Creating the installation configuration file	3205
23.3.11.1. Installation configuration parameters	3206
23.3.11.1.1. Required configuration parameters	3207
23.3.11.1.2. Network configuration parameters	3208
23.3.11.1.3. Optional configuration parameters	3210
23.3.11.1.4. Additional VMware vSphere configuration parameters	3214
23.3.11.1.5. Optional VMware vSphere machine pool configuration parameters	3216
23.3.11.1.6. Region and zone enablement configuration parameters	3217
23.3.11.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3218
23.3.11.3. Configuring the cluster-wide proxy during installation	3220
23.3.11.4. Configuring regions and zones for a VMware vCenter	3222
23.3.12. Deploying the cluster	3225
23.3.13. Installing the OpenShift CLI by downloading the binary	3226
Installing the OpenShift CLI on Linux	3227
Installing the OpenShift CLI on Windows	3227
Installing the OpenShift CLI on macOS	3228
23.3.14. Logging in to the cluster by using the CLI	3228
23.3.15. Creating registry storage	3229
23.3.15.1. Image registry removed during installation	3229
23.3.15.2. Image registry storage configuration	3229
23.3.15.2.1. Configuring registry storage for VMware vSphere	3229
23.3.15.2.2. Configuring block registry storage for VMware vSphere	3231
23.3.16. Backing up VMware vSphere volumes	3232
23.3.17. Telemetry access for OpenShift Container Platform	3233
23.3.18. Services for an external load balancer	3233
23.3.18.1. Configuring an external load balancer	3236
23.3.19. Next steps	3241
23.4. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS	3242
23.4.1. Prerequisites	3242
23.4.2. Internet access for OpenShift Container Platform	3242
23.4.3. VMware vSphere infrastructure requirements	3243
23.4.4. Network connectivity requirements	3244
23.4.5. VMware vSphere CSI Driver Operator requirements	3245
23.4.6. vCenter requirements	3246
Required vCenter account privileges	3246
Using OpenShift Container Platform with vMotion	3255
Cluster resources	3255
Cluster limits	3256
Networking requirements	3256
Required IP Addresses	3256
DNS records	3256
23.4.7. Generating a key pair for cluster node SSH access	3257
23.4.8. Obtaining the installation program	3259
23.4.9. Adding vCenter root CA certificates to your system trust	3260

23.4.10. VMware vSphere region and zone enablement	3260
23.4.11. Creating the installation configuration file	3262
23.4.11.1. Installation configuration parameters	3263
23.4.11.1.1. Required configuration parameters	3264
23.4.11.1.2. Network configuration parameters	3265
23.4.11.1.3. Optional configuration parameters	3267
23.4.11.1.4. Additional VMware vSphere configuration parameters	3271
23.4.11.1.5. Optional VMware vSphere machine pool configuration parameters	3273
23.4.11.1.6. Region and zone enablement configuration parameters	3274
23.4.11.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3275
23.4.11.3. Configuring the cluster-wide proxy during installation	3277
23.4.11.4. Configuring regions and zones for a VMware vCenter	3279
23.4.12. Network configuration phases	3282
23.4.13. Specifying advanced network configuration	3283
23.4.14. Cluster Network Operator configuration	3284
23.4.14.1. Cluster Network Operator configuration object	3284
defaultNetwork object configuration	3285
Configuration for the OpenShift SDN network plugin	3286
Configuration for the OVN-Kubernetes network plugin	3287
kubeProxyConfig object configuration	3291
23.4.15. Deploying the cluster	3292
23.4.16. Installing the OpenShift CLI by downloading the binary	3294
Installing the OpenShift CLI on Linux	3294
Installing the OpenShift CLI on Windows	3294
Installing the OpenShift CLI on macOS	3295
23.4.17. Logging in to the cluster by using the CLI	3295
23.4.18. Creating registry storage	3296
23.4.18.1. Image registry removed during installation	3296
23.4.18.2. Image registry storage configuration	3296
23.4.18.2.1. Configuring registry storage for VMware vSphere	3296
23.4.18.2.2. Configuring block registry storage for VMware vSphere	3298
23.4.19. Backing up VMware vSphere volumes	3299
23.4.20. Telemetry access for OpenShift Container Platform	3300
23.4.21. Services for an external load balancer	3300
23.4.21.1. Configuring an external load balancer	3303
23.4.22. Next steps	3308
23.5. INSTALLING A CLUSTER ON VSPHERE WITH USER-PROVISIONED INFRASTRUCTURE	3309
23.5.1. Prerequisites	3309
23.5.2. Internet access for OpenShift Container Platform	3309
23.5.3. VMware vSphere infrastructure requirements	3310
23.5.4. VMware vSphere CSI Driver Operator requirements	3311
23.5.5. Requirements for a cluster with user-provisioned infrastructure	3312
23.5.5.1. vCenter requirements	3312
Required vCenter account privileges	3312
Using OpenShift Container Platform with vMotion	3321
Cluster resources	3321
Cluster limits	3321
Networking requirements	3322
Required IP Addresses	3322
DNS records	3322
23.5.5.2. Required machines for cluster installation	3322
23.5.5.3. Minimum resource requirements for cluster installation	3323
23.5.5.4. Certificate signing requests management	3324

23.5.5.5. Networking requirements for user-provisioned infrastructure	3324
23.5.5.5.1. Setting the cluster node hostnames through DHCP	3325
23.5.5.5.2. Network connectivity requirements	3325
Ethernet adaptor hardware address requirements	3326
NTP configuration for user-provisioned infrastructure	3326
23.5.5.6. User-provisioned DNS requirements	3327
23.5.5.6.1. Example DNS configuration for user-provisioned clusters	3328
23.5.5.7. Load balancing requirements for user-provisioned infrastructure	3331
23.5.5.7.1. Example load balancer configuration for user-provisioned clusters	3333
23.5.6. Preparing the user-provisioned infrastructure	3334
23.5.7. Validating DNS resolution for user-provisioned infrastructure	3336
23.5.8. Generating a key pair for cluster node SSH access	3339
23.5.9. VMware vSphere region and zone enablement	3340
23.5.10. Obtaining the installation program	3342
23.5.11. Manually creating the installation configuration file	3343
23.5.11.1. Sample install-config.yaml file for VMware vSphere	3343
23.5.11.2. Configuring the cluster-wide proxy during installation	3346
23.5.11.3. Configuring regions and zones for a VMware vCenter	3347
23.5.12. Creating the Kubernetes manifest and Ignition config files	3351
23.5.13. Extracting the infrastructure name	3352
23.5.14. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	3353
23.5.15. Adding more compute machines to a cluster in vSphere	3357
23.5.16. Disk partitioning	3359
Creating a separate /var partition	3359
23.5.17. Installing the OpenShift CLI by downloading the binary	3361
Installing the OpenShift CLI on Linux	3361
Installing the OpenShift CLI on Windows	3362
Installing the OpenShift CLI on macOS	3362
23.5.18. Waiting for the bootstrap process to complete	3363
23.5.19. Logging in to the cluster by using the CLI	3364
23.5.20. Approving the certificate signing requests for your machines	3364
23.5.21. Initial Operator configuration	3367
23.5.21.1. Image registry removed during installation	3368
23.5.21.2. Image registry storage configuration	3368
23.5.21.2.1. Configuring registry storage for VMware vSphere	3369
23.5.21.2.2. Configuring storage for the image registry in non-production clusters	3370
23.5.21.2.3. Configuring block registry storage for VMware vSphere	3371
23.5.22. Completing installation on user-provisioned infrastructure	3372
23.5.23. Configuring vSphere DRS anti-affinity rules for control plane nodes	3375
23.5.24. Backing up VMware vSphere volumes	3376
23.5.25. Telemetry access for OpenShift Container Platform	3376
23.5.26. Next steps	3376
23.6. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS	3377
23.6.1. Prerequisites	3377
23.6.2. Internet access for OpenShift Container Platform	3377
23.6.3. VMware vSphere infrastructure requirements	3378
23.6.4. VMware vSphere CSI Driver Operator requirements	3379
23.6.5. Requirements for a cluster with user-provisioned infrastructure	3380
23.6.5.1. vCenter requirements	3380
Required vCenter account privileges	3380
Using OpenShift Container Platform with vMotion	3389
Cluster resources	3389
Cluster limits	3390

Networking requirements	3390
Required IP Addresses	3390
DNS records	3390
23.6.5.2. Required machines for cluster installation	3391
23.6.5.3. Minimum resource requirements for cluster installation	3391
23.6.5.4. Certificate signing requests management	3392
23.6.5.5. Networking requirements for user-provisioned infrastructure	3392
23.6.5.5.1. Setting the cluster node hostnames through DHCP	3393
23.6.5.5.2. Network connectivity requirements	3393
Ethernet adaptor hardware address requirements	3394
NTP configuration for user-provisioned infrastructure	3394
23.6.5.6. User-provisioned DNS requirements	3395
23.6.5.6.1. Example DNS configuration for user-provisioned clusters	3397
23.6.5.7. Load balancing requirements for user-provisioned infrastructure	3399
23.6.5.7.1. Example load balancer configuration for user-provisioned clusters	3401
23.6.6. Preparing the user-provisioned infrastructure	3403
23.6.7. Validating DNS resolution for user-provisioned infrastructure	3405
23.6.8. Generating a key pair for cluster node SSH access	3407
23.6.9. VMware vSphere region and zone enablement	3409
23.6.10. Obtaining the installation program	3410
23.6.11. Manually creating the installation configuration file	3411
23.6.11.1. Sample install-config.yaml file for VMware vSphere	3412
23.6.11.2. Configuring the cluster-wide proxy during installation	3414
23.6.11.3. Configuring regions and zones for a VMware vCenter	3416
23.6.12. Network configuration phases	3419
23.6.13. Specifying advanced network configuration	3420
23.6.14. Cluster Network Operator configuration	3421
23.6.14.1. Cluster Network Operator configuration object	3422
defaultNetwork object configuration	3422
Configuration for the OpenShift SDN network plugin	3423
Configuration for the OVN-Kubernetes network plugin	3424
kubeProxyConfig object configuration	3428
23.6.15. Creating the Ignition config files	3429
23.6.16. Extracting the infrastructure name	3430
23.6.17. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	3431
23.6.18. Adding more compute machines to a cluster in vSphere	3435
23.6.19. Disk partitioning	3437
Creating a separate /var partition	3437
23.6.20. Waiting for the bootstrap process to complete	3439
23.6.21. Logging in to the cluster by using the CLI	3440
23.6.22. Approving the certificate signing requests for your machines	3441
23.6.22.1. Initial Operator configuration	3443
23.6.22.2. Image registry removed during installation	3444
23.6.22.3. Image registry storage configuration	3444
23.6.22.3.1. Configuring block registry storage for VMware vSphere	3445
23.6.23. Completing installation on user-provisioned infrastructure	3446
23.6.24. Configuring vSphere DRS anti-affinity rules for control plane nodes	3449
23.6.25. Backing up VMware vSphere volumes	3450
23.6.26. Telemetry access for OpenShift Container Platform	3450
23.6.27. Next steps	3450
23.7. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK	3451
23.7.1. Prerequisites	3451
23.7.2. About installations in restricted networks	3451

23.7.2.1. Additional limits	3452
23.7.3. Internet access for OpenShift Container Platform	3452
23.7.4. VMware vSphere infrastructure requirements	3452
23.7.5. Network connectivity requirements	3454
23.7.6. VMware vSphere CSI Driver Operator requirements	3455
23.7.7. vCenter requirements	3455
Required vCenter account privileges	3455
Using OpenShift Container Platform with vMotion	3464
Cluster resources	3464
Cluster limits	3465
Networking requirements	3465
Required IP Addresses	3465
DNS records	3465
23.7.8. Generating a key pair for cluster node SSH access	3466
23.7.9. Adding vCenter root CA certificates to your system trust	3468
23.7.10. Creating the RHCOS image for restricted network installations	3469
23.7.11. VMware vSphere region and zone enablement	3469
23.7.12. Creating the installation configuration file	3471
23.7.12.1. Installation configuration parameters	3473
23.7.12.1.1. Required configuration parameters	3473
23.7.12.1.2. Network configuration parameters	3475
23.7.12.1.3. Optional configuration parameters	3477
23.7.12.1.4. Additional VMware vSphere configuration parameters	3482
23.7.12.1.5. Optional VMware vSphere machine pool configuration parameters	3483
23.7.12.1.6. Region and zone enablement configuration parameters	3484
23.7.12.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3486
23.7.12.3. Configuring the cluster-wide proxy during installation	3487
23.7.12.4. Configuring regions and zones for a VMware vCenter	3489
23.7.13. Deploying the cluster	3492
23.7.14. Installing the OpenShift CLI by downloading the binary	3494
Installing the OpenShift CLI on Linux	3494
Installing the OpenShift CLI on Windows	3495
Installing the OpenShift CLI on macOS	3495
23.7.15. Logging in to the cluster by using the CLI	3496
23.7.16. Disabling the default OperatorHub catalog sources	3496
23.7.17. Creating registry storage	3497
23.7.17.1. Image registry removed during installation	3497
23.7.17.2. Image registry storage configuration	3497
23.7.17.2.1. Configuring registry storage for VMware vSphere	3497
23.7.18. Telemetry access for OpenShift Container Platform	3499
23.7.19. Services for an external load balancer	3499
23.7.19.1. Configuring an external load balancer	3502
23.7.20. Next steps	3507
23.8. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE	3508
23.8.1. Prerequisites	3508
23.8.2. About installations in restricted networks	3509
23.8.2.1. Additional limits	3509
23.8.3. Internet access for OpenShift Container Platform	3509
23.8.4. VMware vSphere infrastructure requirements	3510
23.8.5. VMware vSphere CSI Driver Operator requirements	3511
23.8.6. Requirements for a cluster with user-provisioned infrastructure	3512
23.8.6.1. vCenter requirements	3512

Required vCenter account privileges	3512
Using OpenShift Container Platform with vMotion	3521
Cluster resources	3521
Cluster limits	3521
Networking requirements	3522
Required IP Addresses	3522
DNS records	3522
23.8.6.2. Required machines for cluster installation	3522
23.8.6.3. Minimum resource requirements for cluster installation	3523
23.8.6.4. Certificate signing requests management	3524
23.8.6.5. Networking requirements for user-provisioned infrastructure	3524
23.8.6.5.1. Setting the cluster node hostnames through DHCP	3525
23.8.6.5.2. Network connectivity requirements	3525
Ethernet adaptor hardware address requirements	3526
NTP configuration for user-provisioned infrastructure	3526
23.8.6.6. User-provisioned DNS requirements	3527
23.8.6.6.1. Example DNS configuration for user-provisioned clusters	3528
23.8.6.7. Load balancing requirements for user-provisioned infrastructure	3531
23.8.6.7.1. Example load balancer configuration for user-provisioned clusters	3533
23.8.7. Preparing the user-provisioned infrastructure	3534
23.8.8. Validating DNS resolution for user-provisioned infrastructure	3536
23.8.9. Generating a key pair for cluster node SSH access	3539
23.8.10. VMware vSphere region and zone enablement	3540
23.8.11. Manually creating the installation configuration file	3542
23.8.11.1. Sample install-config.yaml file for VMware vSphere	3543
23.8.11.2. Configuring the cluster-wide proxy during installation	3545
23.8.11.3. Configuring regions and zones for a VMware vCenter	3547
23.8.12. Creating the Kubernetes manifest and Ignition config files	3550
23.8.13. Configuring chrony time service	3552
23.8.14. Extracting the infrastructure name	3553
23.8.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	3554
23.8.16. Adding more compute machines to a cluster in vSphere	3558
23.8.17. Disk partitioning	3560
Creating a separate /var partition	3560
23.8.18. Waiting for the bootstrap process to complete	3562
23.8.19. Logging in to the cluster by using the CLI	3563
23.8.20. Approving the certificate signing requests for your machines	3564
23.8.21. Initial Operator configuration	3566
23.8.21.1. Disabling the default OperatorHub catalog sources	3567
23.8.21.2. Image registry storage configuration	3568
23.8.21.2.1. Configuring registry storage for VMware vSphere	3568
23.8.21.2.2. Configuring storage for the image registry in non-production clusters	3569
23.8.21.2.3. Configuring block registry storage for VMware vSphere	3570
23.8.22. Completing installation on user-provisioned infrastructure	3571
23.8.23. Configuring vSphere DRS anti-affinity rules for control plane nodes	3574
23.8.24. Backing up VMware vSphere volumes	3575
23.8.25. Telemetry access for OpenShift Container Platform	3575
23.8.26. Next steps	3575
23.9. CONFIGURING THE VSPHERE CONNECTION SETTINGS AFTER AN INSTALLATION	3576
23.9.1. Configuring the vSphere connection settings	3576
23.9.2. Verifying the configuration	3578
23.10. UNINSTALLING A CLUSTER ON VSPHERE THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE	3579

23.10.1. Removing a cluster that uses installer-provisioned infrastructure	3579
23.11. USING THE VSPHERE PROBLEM DETECTOR OPERATOR	3580
23.11.1. About the vSphere Problem Detector Operator	3580
23.11.2. Running the vSphere Problem Detector Operator checks	3580
23.11.3. Viewing the events from the vSphere Problem Detector Operator	3581
23.11.4. Viewing the logs from the vSphere Problem Detector Operator	3582
23.11.5. Configuration checks run by the vSphere Problem Detector Operator	3582
23.11.6. About the storage class configuration check	3584
23.11.7. Metrics for the vSphere Problem Detector Operator	3584
23.11.8. Additional resources	3585
CHAPTER 24. INSTALLING ON VMC	3586
24.1. PREPARING TO INSTALL ON VMC	3586
24.1.1. Prerequisites	3586
24.1.2. Choosing a method to install OpenShift Container Platform on VMC	3586
24.1.2.1. Installer-provisioned infrastructure installation of OpenShift Container Platform on VMC	3586
24.1.2.2. User-provisioned infrastructure installation of OpenShift Container Platform on VMC	3587
24.1.3. VMware vSphere infrastructure requirements	3587
24.1.4. VMware vSphere CSI Driver Operator requirements	3588
24.1.5. Uninstalling an installer-provisioned infrastructure installation of OpenShift Container Platform on VMC	3588
24.2. INSTALLING A CLUSTER ON VMC	3589
24.2.1. Setting up VMC for vSphere	3589
24.2.1.1. VMC Sizer tool	3591
24.2.2. vSphere prerequisites	3591
24.2.3. Internet access for OpenShift Container Platform	3591
24.2.4. VMware vSphere infrastructure requirements	3592
24.2.5. Network connectivity requirements	3593
24.2.6. VMware vSphere CSI Driver Operator requirements	3594
24.2.7. vCenter requirements	3595
Required vCenter account privileges	3595
Using OpenShift Container Platform with vMotion	3604
Cluster resources	3604
Cluster limits	3605
Networking requirements	3605
Required IP Addresses	3605
DNS records	3605
24.2.8. Generating a key pair for cluster node SSH access	3606
24.2.9. Obtaining the installation program	3608
24.2.10. Adding vCenter root CA certificates to your system trust	3609
24.2.11. Deploying the cluster	3609
24.2.12. Installing the OpenShift CLI by downloading the binary	3612
Installing the OpenShift CLI on Linux	3613
Installing the OpenShift CLI on Windows	3613
Installing the OpenShift CLI on macOS	3614
24.2.13. Logging in to the cluster by using the CLI	3614
24.2.14. Creating registry storage	3615
24.2.14.1. Image registry removed during installation	3615
24.2.14.2. Image registry storage configuration	3615
24.2.14.2.1. Configuring registry storage for VMware vSphere	3615
24.2.14.2.2. Configuring block registry storage for VMware vSphere	3617
24.2.15. Backing up VMware vSphere volumes	3618
24.2.16. Telemetry access for OpenShift Container Platform	3619

24.2.17. Services for an external load balancer	3619
24.2.17.1. Configuring an external load balancer	3622
24.2.18. Next steps	3627
24.3. INSTALLING A CLUSTER ON VMC WITH CUSTOMIZATIONS	3628
24.3.1. Setting up VMC for vSphere	3628
24.3.1.1. VMC Sizer tool	3630
24.3.2. vSphere prerequisites	3630
24.3.3. Internet access for OpenShift Container Platform	3631
24.3.4. VMware vSphere infrastructure requirements	3631
24.3.5. Network connectivity requirements	3632
24.3.6. VMware vSphere CSI Driver Operator requirements	3633
24.3.7. vCenter requirements	3634
Required vCenter account privileges	3634
Using OpenShift Container Platform with vMotion	3643
Cluster resources	3643
Cluster limits	3644
Networking requirements	3644
Required IP Addresses	3644
DNS records	3644
24.3.8. Generating a key pair for cluster node SSH access	3645
24.3.9. Obtaining the installation program	3647
24.3.10. Adding vCenter root CA certificates to your system trust	3648
24.3.11. VMware vSphere region and zone enablement	3648
24.3.12. Creating the installation configuration file	3650
24.3.12.1. Installation configuration parameters	3651
24.3.12.1.1. Required configuration parameters	3652
24.3.12.1.2. Network configuration parameters	3653
24.3.12.1.3. Optional configuration parameters	3655
24.3.12.1.4. Additional VMware vSphere configuration parameters	3660
24.3.12.1.5. Optional VMware vSphere machine pool configuration parameters	3662
24.3.12.1.6. Region and zone enablement configuration parameters	3663
24.3.12.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3664
24.3.12.3. Configuring the cluster-wide proxy during installation	3666
24.3.12.4. Configuring regions and zones for a VMware vCenter	3668
24.3.13. Deploying the cluster	3671
24.3.14. Installing the OpenShift CLI by downloading the binary	3673
Installing the OpenShift CLI on Linux	3673
Installing the OpenShift CLI on Windows	3674
Installing the OpenShift CLI on macOS	3674
24.3.15. Logging in to the cluster by using the CLI	3675
24.3.16. Creating registry storage	3675
24.3.16.1. Image registry removed during installation	3675
24.3.16.2. Image registry storage configuration	3675
24.3.16.2.1. Configuring registry storage for VMware vSphere	3676
24.3.16.2.2. Configuring block registry storage for VMware vSphere	3677
24.3.17. Backing up VMware vSphere volumes	3679
24.3.18. Telemetry access for OpenShift Container Platform	3679
24.3.19. Services for an external load balancer	3679
24.3.19.1. Configuring an external load balancer	3682
24.3.20. Next steps	3687
24.4. INSTALLING A CLUSTER ON VMC WITH NETWORK CUSTOMIZATIONS	3688
24.4.1. Setting up VMC for vSphere	3688
24.4.1.1. VMC Sizer tool	3690

24.4.2. vSphere prerequisites	3690
24.4.3. Internet access for OpenShift Container Platform	3691
24.4.4. VMware vSphere infrastructure requirements	3691
24.4.5. Network connectivity requirements	3692
24.4.6. VMware vSphere CSI Driver Operator requirements	3693
24.4.7. vCenter requirements	3694
Required vCenter account privileges	3694
Using OpenShift Container Platform with vMotion	3703
Cluster resources	3703
Cluster limits	3704
Networking requirements	3704
Required IP Addresses	3704
DNS records	3704
24.4.8. Generating a key pair for cluster node SSH access	3705
24.4.9. Obtaining the installation program	3707
24.4.10. Adding vCenter root CA certificates to your system trust	3708
24.4.11. VMware vSphere region and zone enablement	3708
24.4.12. Creating the installation configuration file	3710
24.4.12.1. Installation configuration parameters	3711
24.4.12.1.1. Required configuration parameters	3712
24.4.12.1.2. Network configuration parameters	3713
24.4.12.1.3. Optional configuration parameters	3715
24.4.12.1.4. Additional VMware vSphere configuration parameters	3720
24.4.12.1.5. Optional VMware vSphere machine pool configuration parameters	3722
24.4.12.1.6. Region and zone enablement configuration parameters	3723
24.4.12.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3724
24.4.12.3. Configuring the cluster-wide proxy during installation	3726
24.4.12.4. Configuring regions and zones for a VMware vCenter	3728
24.4.13. Network configuration phases	3731
24.4.14. Specifying advanced network configuration	3732
24.4.15. Cluster Network Operator configuration	3733
24.4.15.1. Cluster Network Operator configuration object	3734
defaultNetwork object configuration	3735
Configuration for the OpenShift SDN network plugin	3735
Configuration for the OVN-Kubernetes network plugin	3736
kubeProxyConfig object configuration	3740
24.4.16. Deploying the cluster	3741
24.4.17. Installing the OpenShift CLI by downloading the binary	3743
Installing the OpenShift CLI on Linux	3743
Installing the OpenShift CLI on Windows	3744
Installing the OpenShift CLI on macOS	3744
24.4.18. Logging in to the cluster by using the CLI	3745
24.4.19. Creating registry storage	3745
24.4.19.1. Image registry removed during installation	3745
24.4.19.2. Image registry storage configuration	3745
24.4.19.2.1. Configuring registry storage for VMware vSphere	3746
24.4.19.2.2. Configuring block registry storage for VMware vSphere	3747
24.4.20. Backing up VMware vSphere volumes	3749
24.4.21. Telemetry access for OpenShift Container Platform	3749
24.4.22. Services for an external load balancer	3749
24.4.22.1. Configuring an external load balancer	3752
24.4.23. Next steps	3757
24.5. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK	3758

24.5.1. Setting up VMC for vSphere	3758
24.5.1.1. VMC Sizer tool	3760
24.5.2. vSphere prerequisites	3760
24.5.3. About installations in restricted networks	3761
24.5.3.1. Additional limits	3761
24.5.4. Internet access for OpenShift Container Platform	3761
24.5.5. VMware vSphere infrastructure requirements	3762
24.5.6. Network connectivity requirements	3763
24.5.7. VMware vSphere CSI Driver Operator requirements	3764
24.5.8. vCenter requirements	3765
Required vCenter account privileges	3765
Using OpenShift Container Platform with vMotion	3774
Cluster resources	3774
Cluster limits	3775
Networking requirements	3775
Required IP Addresses	3775
DNS records	3775
24.5.9. Generating a key pair for cluster node SSH access	3776
24.5.10. Adding vCenter root CA certificates to your system trust	3778
24.5.11. Creating the RHCOS image for restricted network installations	3778
24.5.12. VMware vSphere region and zone enablement	3779
24.5.13. Creating the installation configuration file	3780
24.5.13.1. Installation configuration parameters	3783
24.5.13.1.1. Required configuration parameters	3783
24.5.13.1.2. Network configuration parameters	3785
24.5.13.1.3. Optional configuration parameters	3787
24.5.13.1.4. Additional VMware vSphere configuration parameters	3792
24.5.13.1.5. Optional VMware vSphere machine pool configuration parameters	3794
24.5.13.1.6. Region and zone enablement configuration parameters	3795
24.5.13.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster	3796
24.5.13.3. Configuring the cluster-wide proxy during installation	3798
24.5.13.4. Configuring regions and zones for a VMware vCenter	3800
24.5.14. Deploying the cluster	3803
24.5.15. Installing the OpenShift CLI by downloading the binary	3805
Installing the OpenShift CLI on Linux	3805
Installing the OpenShift CLI on Windows	3806
Installing the OpenShift CLI on macOS	3806
24.5.16. Logging in to the cluster by using the CLI	3807
24.5.17. Disabling the default OperatorHub catalog sources	3807
24.5.18. Creating registry storage	3808
24.5.18.1. Image registry removed during installation	3808
24.5.18.2. Image registry storage configuration	3808
24.5.18.2.1. Configuring registry storage for VMware vSphere	3808
24.5.19. Telemetry access for OpenShift Container Platform	3810
24.5.20. Services for an external load balancer	3810
24.5.20.1. Configuring an external load balancer	3813
24.5.21. Next steps	3819
24.6. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE	3819
24.6.1. Setting up VMC for vSphere	3819
24.6.1.1. VMC Sizer tool	3821
24.6.2. vSphere prerequisites	3822
24.6.3. Internet access for OpenShift Container Platform	3822
24.6.4. VMware vSphere infrastructure requirements	3822

24.6.5. VMware vSphere CSI Driver Operator requirements	3823
24.6.6. Requirements for a cluster with user-provisioned infrastructure	3824
24.6.6.1. vCenter requirements	3824
Required vCenter account privileges	3824
Using OpenShift Container Platform with vMotion	3833
Cluster resources	3833
Cluster limits	3834
Networking requirements	3834
Required IP Addresses	3834
DNS records	3834
24.6.6.2. Required machines for cluster installation	3835
24.6.6.3. Minimum resource requirements for cluster installation	3835
24.6.6.4. Certificate signing requests management	3836
24.6.6.5. Networking requirements for user-provisioned infrastructure	3836
24.6.6.5.1. Setting the cluster node hostnames through DHCP	3837
24.6.6.5.2. Network connectivity requirements	3837
Ethernet adaptor hardware address requirements	3838
NTP configuration for user-provisioned infrastructure	3838
24.6.6.6. User-provisioned DNS requirements	3839
24.6.6.6.1. Example DNS configuration for user-provisioned clusters	3840
24.6.6.7. Load balancing requirements for user-provisioned infrastructure	3843
24.6.6.7.1. Example load balancer configuration for user-provisioned clusters	3844
24.6.7. Preparing the user-provisioned infrastructure	3846
24.6.8. Validating DNS resolution for user-provisioned infrastructure	3848
24.6.9. Generating a key pair for cluster node SSH access	3850
24.6.10. VMware vSphere region and zone enablement	3852
24.6.11. Obtaining the installation program	3853
24.6.12. Manually creating the installation configuration file	3854
24.6.12.1. Sample install-config.yaml file for VMware vSphere	3855
24.6.12.2. Configuring the cluster-wide proxy during installation	3857
24.6.12.3. Configuring regions and zones for a VMware vCenter	3859
24.6.13. Creating the Kubernetes manifest and Ignition config files	3862
24.6.14. Extracting the infrastructure name	3864
24.6.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	3865
24.6.16. Adding more compute machines to a cluster in vSphere	3869
24.6.17. Disk partitioning	3870
Creating a separate /var partition	3871
24.6.18. Installing the OpenShift CLI by downloading the binary	3873
Installing the OpenShift CLI on Linux	3873
Installing the OpenShift CLI on Windows	3874
Installing the OpenShift CLI on macOS	3874
24.6.19. Waiting for the bootstrap process to complete	3875
24.6.20. Logging in to the cluster by using the CLI	3876
24.6.21. Approving the certificate signing requests for your machines	3876
24.6.22. Initial Operator configuration	3879
24.6.22.1. Image registry removed during installation	3880
24.6.22.2. Image registry storage configuration	3880
24.6.22.2.1. Configuring registry storage for VMware vSphere	3880
24.6.22.2.2. Configuring storage for the image registry in non-production clusters	3882
24.6.22.2.3. Configuring block registry storage for VMware vSphere	3883
24.6.23. Completing installation on user-provisioned infrastructure	3884
24.6.24. Backing up VMware vSphere volumes	3886
24.6.25. Telemetry access for OpenShift Container Platform	3887

24.6.26. Next steps	3887
24.7. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE AND NETWORK CUSTOMIZATIONS	3887
24.7.1. Setting up VMC for vSphere	3888
24.7.1.1. VMC Sizer tool	3889
24.7.2. vSphere prerequisites	3890
24.7.3. Internet access for OpenShift Container Platform	3890
24.7.4. VMware vSphere infrastructure requirements	3890
24.7.5. VMware vSphere CSI Driver Operator requirements	3891
24.7.6. Requirements for a cluster with user-provisioned infrastructure	3892
24.7.6.1. vCenter requirements	3892
Required vCenter account privileges	3892
Using OpenShift Container Platform with vMotion	3901
Cluster resources	3901
Cluster limits	3902
Networking requirements	3902
Required IP Addresses	3902
DNS records	3902
24.7.6.2. Required machines for cluster installation	3903
24.7.6.3. Minimum resource requirements for cluster installation	3903
24.7.6.4. Certificate signing requests management	3904
24.7.6.5. Networking requirements for user-provisioned infrastructure	3904
24.7.6.5.1. Setting the cluster node hostnames through DHCP	3905
24.7.6.5.2. Network connectivity requirements	3905
Ethernet adaptor hardware address requirements	3906
NTP configuration for user-provisioned infrastructure	3906
24.7.6.6. User-provisioned DNS requirements	3907
24.7.6.6.1. Example DNS configuration for user-provisioned clusters	3908
24.7.6.7. Load balancing requirements for user-provisioned infrastructure	3911
24.7.6.7.1. Example load balancer configuration for user-provisioned clusters	3912
24.7.7. Preparing the user-provisioned infrastructure	3914
24.7.8. Validating DNS resolution for user-provisioned infrastructure	3916
24.7.9. Generating a key pair for cluster node SSH access	3918
24.7.10. VMware vSphere region and zone enablement	3920
24.7.11. Obtaining the installation program	3921
24.7.12. Manually creating the installation configuration file	3922
24.7.12.1. Sample install-config.yaml file for VMware vSphere	3923
24.7.12.2. Configuring the cluster-wide proxy during installation	3925
24.7.12.3. Configuring regions and zones for a VMware vCenter	3927
24.7.13. Specifying advanced network configuration	3930
24.7.14. Cluster Network Operator configuration	3931
24.7.14.1. Cluster Network Operator configuration object	3932
defaultNetwork object configuration	3933
Configuration for the OpenShift SDN network plugin	3933
Configuration for the OVN-Kubernetes network plugin	3934
kubeProxyConfig object configuration	3938
24.7.15. Creating the Ignition config files	3939
24.7.16. Extracting the infrastructure name	3940
24.7.17. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	3941
24.7.18. Adding more compute machines to a cluster in vSphere	3945
24.7.19. Disk partitioning	3947
Creating a separate /var partition	3947
24.7.20. Waiting for the bootstrap process to complete	3949

24.7.21. Logging in to the cluster by using the CLI	3950
24.7.22. Approving the certificate signing requests for your machines	3951
24.7.23. Initial Operator configuration	3953
24.7.23.1. Image registry removed during installation	3954
24.7.23.2. Image registry storage configuration	3954
24.7.23.2.1. Configuring block registry storage for VMware vSphere	3955
24.7.24. Completing installation on user-provisioned infrastructure	3956
24.7.25. Backing up VMware vSphere volumes	3959
24.7.26. Telemetry access for OpenShift Container Platform	3959
24.7.27. Next steps	3959
24.8. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE	3959
24.8.1. Setting up VMC for vSphere	3960
24.8.1.1. VMC Sizer tool	3961
24.8.2. vSphere prerequisites	3962
24.8.3. About installations in restricted networks	3962
24.8.3.1. Additional limits	3963
24.8.4. Internet access for OpenShift Container Platform	3963
24.8.5. VMware vSphere infrastructure requirements	3963
24.8.6. VMware vSphere CSI Driver Operator requirements	3964
24.8.7. Requirements for a cluster with user-provisioned infrastructure	3965
24.8.7.1. vCenter requirements	3965
Required vCenter account privileges	3965
Using OpenShift Container Platform with vMotion	3974
Cluster resources	3974
Cluster limits	3975
Networking requirements	3975
Required IP Addresses	3975
DNS records	3975
24.8.7.2. Required machines for cluster installation	3976
24.8.7.3. Minimum resource requirements for cluster installation	3976
24.8.7.4. Certificate signing requests management	3977
24.8.7.5. Networking requirements for user-provisioned infrastructure	3977
24.8.7.5.1. Setting the cluster node hostnames through DHCP	3978
24.8.7.5.2. Network connectivity requirements	3978
Ethernet adaptor hardware address requirements	3979
NTP configuration for user-provisioned infrastructure	3979
24.8.7.6. User-provisioned DNS requirements	3980
24.8.7.6.1. Example DNS configuration for user-provisioned clusters	3981
24.8.7.7. Load balancing requirements for user-provisioned infrastructure	3984
24.8.7.7.1. Example load balancer configuration for user-provisioned clusters	3985
24.8.8. Preparing the user-provisioned infrastructure	3987
24.8.9. Validating DNS resolution for user-provisioned infrastructure	3989
24.8.10. Generating a key pair for cluster node SSH access	3991
24.8.11. VMware vSphere region and zone enablement	3993
24.8.12. Manually creating the installation configuration file	3994
24.8.12.1. Sample install-config.yaml file for VMware vSphere	3995
24.8.12.2. Configuring the cluster-wide proxy during installation	3998
24.8.12.3. Configuring regions and zones for a VMware vCenter	3999
24.8.13. Creating the Kubernetes manifest and Ignition config files	4003
24.8.14. Extracting the infrastructure name	4004
24.8.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	4005
24.8.16. Adding more compute machines to a cluster in vSphere	4010

24.8.17. Disk partitioning	4011
Creating a separate /var partition	4011
24.8.18. Waiting for the bootstrap process to complete	4014
24.8.19. Logging in to the cluster by using the CLI	4014
24.8.20. Approving the certificate signing requests for your machines	4015
24.8.21. Initial Operator configuration	4018
24.8.21.1. Disabling the default OperatorHub catalog sources	4019
24.8.21.2. Image registry storage configuration	4019
24.8.21.2.1. Configuring registry storage for VMware vSphere	4019
24.8.21.2.2. Configuring storage for the image registry in non-production clusters	4021
24.8.21.2.3. Configuring block registry storage for VMware vSphere	4021
24.8.22. Completing installation on user-provisioned infrastructure	4023
24.8.23. Backing up VMware vSphere volumes	4025
24.8.24. Telemetry access for OpenShift Container Platform	4025
24.8.25. Next steps	4026
24.9. UNINSTALLING A CLUSTER ON VMC	4026
24.9.1. Removing a cluster that uses installer-provisioned infrastructure	4026
CHAPTER 25. INSTALLING ON ANY PLATFORM	4028
25.1. INSTALLING A CLUSTER ON ANY PLATFORM	4028
25.1.1. Prerequisites	4028
25.1.2. Internet access for OpenShift Container Platform	4028
25.1.3. Requirements for a cluster with user-provisioned infrastructure	4028
25.1.3.1. Required machines for cluster installation	4029
25.1.3.2. Minimum resource requirements for cluster installation	4029
25.1.3.3. Certificate signing requests management	4030
25.1.3.4. Networking requirements for user-provisioned infrastructure	4030
25.1.3.4.1. Setting the cluster node hostnames through DHCP	4031
25.1.3.4.2. Network connectivity requirements	4031
NTP configuration for user-provisioned infrastructure	4032
25.1.3.5. User-provisioned DNS requirements	4033
25.1.3.5.1. Example DNS configuration for user-provisioned clusters	4034
25.1.3.6. Load balancing requirements for user-provisioned infrastructure	4037
25.1.3.6.1. Example load balancer configuration for user-provisioned clusters	4038
25.1.4. Preparing the user-provisioned infrastructure	4040
25.1.5. Validating DNS resolution for user-provisioned infrastructure	4042
25.1.6. Generating a key pair for cluster node SSH access	4044
25.1.7. Obtaining the installation program	4046
25.1.8. Installing the OpenShift CLI by downloading the binary	4047
Installing the OpenShift CLI on Linux	4047
Installing the OpenShift CLI on Windows	4048
Installing the OpenShift CLI on macOS	4048
25.1.9. Manually creating the installation configuration file	4049
25.1.9.1. Sample install-config.yaml file for other platforms	4050
25.1.9.2. Configuring the cluster-wide proxy during installation	4052
25.1.9.3. Configuring a three-node cluster	4054
25.1.10. Creating the Kubernetes manifest and Ignition config files	4055
25.1.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process	4056
25.1.11.1. Installing RHCOS by using an ISO image	4057
25.1.11.2. Installing RHCOS by using PXE or iPXE booting	4061
25.1.11.3. Advanced RHCOS installation configuration	4065
25.1.11.3.1. Using advanced networking options for PXE and ISO installations	4065
25.1.11.3.2. Disk partitioning	4066

25.1.11.3.2.1. Creating a separate /var partition	4067
25.1.11.3.2.2. Retaining existing partitions	4069
25.1.11.3.3. Identifying Ignition configs	4070
25.1.11.3.4. Advanced RHCOS installation reference	4071
25.1.11.3.4.1. Networking and bonding options for ISO installations	4071
Configuring DHCP or static IP addresses	4071
Configuring an IP address without a static hostname	4072
Specifying multiple network interfaces	4072
Configuring default gateway and route	4072
Disabling DHCP on a single interface	4073
Combining DHCP and static IP configurations	4073
Configuring VLANs on individual interfaces	4073
Providing multiple DNS servers	4073
Bonding multiple network interfaces to a single interface	4073
Bonding multiple network interfaces to a single interface	4074
Using network teaming	4074
25.1.11.3.4.2. coreos-installer options for ISO and PXE installations	4074
25.1.11.3.4.3. coreos.inst boot options for ISO or PXE installations	4079
25.1.12. Waiting for the bootstrap process to complete	4080
25.1.13. Logging in to the cluster by using the CLI	4081
25.1.14. Approving the certificate signing requests for your machines	4082
25.1.15. Initial Operator configuration	4085
25.1.15.1. Disabling the default OperatorHub catalog sources	4086
25.1.15.2. Image registry removed during installation	4086
25.1.15.3. Image registry storage configuration	4086
25.1.15.3.1. Configuring registry storage for bare metal and other manual installations	4086
25.1.15.3.2. Configuring storage for the image registry in non-production clusters	4088
25.1.15.3.3. Configuring block registry storage for bare metal	4088
25.1.16. Completing installation on user-provisioned infrastructure	4090
25.1.17. Telemetry access for OpenShift Container Platform	4092
25.1.18. Next steps	4092
CHAPTER 26. INSTALLATION CONFIGURATION	4093
26.1. CUSTOMIZING NODES	4093
26.1.1. Creating machine configs with Butane	4093
26.1.1.1. About Butane	4093
26.1.1.2. Installing Butane	4093
26.1.1.3. Creating a MachineConfig object by using Butane	4094
26.1.2. Adding day-1 kernel arguments	4095
26.1.3. Adding kernel modules to nodes	4096
26.1.3.1. Building and testing the kernel module container	4097
26.1.3.2. Provisioning a kernel module to OpenShift Container Platform	4100
26.1.3.2.1. Provision kernel modules via a MachineConfig object	4100
26.1.4. Encrypting and mirroring disks during installation	4102
26.1.4.1. About disk encryption	4102
26.1.4.1.1. Configuring an encryption threshold	4103
26.1.4.2. About disk mirroring	4104
26.1.4.3. Configuring disk encryption and mirroring	4105
26.1.4.4. Configuring a RAID-enabled data volume	4112
26.1.5. Configuring chrony time service	4114
26.1.6. Additional resources	4115
26.2. CONFIGURING YOUR FIREWALL	4115
26.2.1. Configuring your firewall for OpenShift Container Platform	4115

26.3. ENABLING LINUX CONTROL GROUP VERSION 2 (CGROUP V2)	4120
26.3.1. Enabling Linux cgroup v2 during installation	4120
CHAPTER 27. VALIDATING AN INSTALLATION	4122
27.1. REVIEWING THE INSTALLATION LOG	4122
27.2. VIEWING THE IMAGE PULL SOURCE	4122
27.3. GETTING CLUSTER VERSION, STATUS, AND UPDATE DETAILS	4123
27.4. QUERYING THE STATUS OF THE CLUSTER NODES BY USING THE CLI	4125
27.5. REVIEWING THE CLUSTER STATUS FROM THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE	4125
27.6. REVIEWING THE CLUSTER STATUS FROM RED HAT OPENSIFT CLUSTER MANAGER	4126
27.7. CHECKING CLUSTER RESOURCE AVAILABILITY AND UTILIZATION	4127
27.8. LISTING ALERTS THAT ARE FIRING	4129
27.9. NEXT STEPS	4129
CHAPTER 28. TROUBLESHOOTING INSTALLATION ISSUES	4130
28.1. PREREQUISITES	4130
28.2. GATHERING LOGS FROM A FAILED INSTALLATION	4130
28.3. MANUALLY GATHERING LOGS WITH SSH ACCESS TO YOUR HOST(S)	4131
28.4. MANUALLY GATHERING LOGS WITHOUT SSH ACCESS TO YOUR HOST(S)	4132
28.5. GETTING DEBUG INFORMATION FROM THE INSTALLATION PROGRAM	4132
28.6. REINSTALLING THE OPENSIFT CONTAINER PLATFORM CLUSTER	4133
CHAPTER 29. SUPPORT FOR FIPS CRYPTOGRAPHY	4134
29.1. FIPS VALIDATION IN OPENSIFT CONTAINER PLATFORM	4134
29.2. FIPS SUPPORT IN COMPONENTS THAT THE CLUSTER USES	4135
29.2.1. etcd	4135
29.2.2. Storage	4135
29.2.3. Runtimes	4135
29.3. INSTALLING A CLUSTER IN FIPS MODE	4135

CHAPTER 1. OPENSIFT CONTAINER PLATFORM INSTALLATION OVERVIEW

1.1. ABOUT OPENSIFT CONTAINER PLATFORM INSTALLATION

The OpenShift Container Platform installation program offers four methods for deploying a cluster which are detailed in the following list:

- **Interactive:** You can deploy a cluster with the web-based [Assisted Installer](#). This is an ideal approach for clusters with networks connected to the internet. The Assisted Installer is the easiest way to install OpenShift Container Platform, it provides smart defaults, and it performs pre-flight validations before installing the cluster. It also provides a RESTful API for automation and advanced configuration scenarios.
- **Local Agent-based:** You can deploy a cluster locally with the Agent-based Installer for disconnected environments or restricted networks. It provides many of the benefits of the Assisted Installer, but you must download and configure the [Agent-based Installer](#) first. Configuration is done with a command-line interface. This approach is ideal for disconnected environments.
- **Automated:** You can deploy a cluster on installer-provisioned infrastructure. The installation program uses each cluster host's baseboard management controller (BMC) for provisioning. You can deploy clusters in connected or disconnected environments.
- **Full control:** You can deploy a cluster on infrastructure that you prepare and maintain, which provides maximum customizability. You can deploy clusters in connected or disconnected environments.

Each method deploys a cluster with the following characteristics:

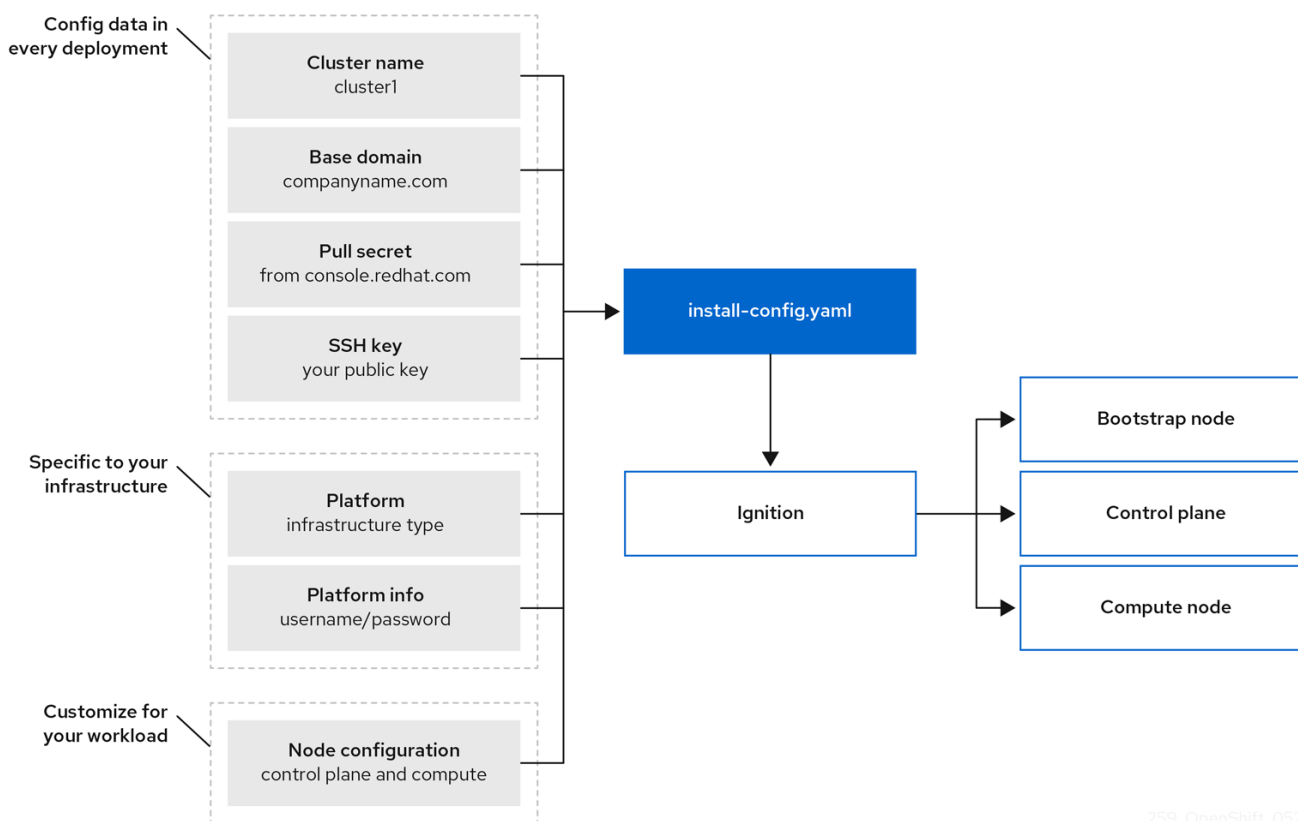
- Highly available infrastructure with no single points of failure, which is available by default.
- Administrators can control what updates are applied and when.

1.1.1. About the installation program

You can use the installation program to deploy each type of cluster. The installation program generates the main assets, such as Ignition config files for the bootstrap, control plane, and compute machines. You can start an OpenShift Container Platform cluster with these three machine configurations, provided you correctly configured the infrastructure.

The OpenShift Container Platform installation program uses a set of targets and dependencies to manage cluster installations. The installation program has a set of targets that it must achieve, and each target has a set of dependencies. Because each target is only concerned with its own dependencies, the installation program can act to achieve multiple targets in parallel with the ultimate target being a running cluster. The installation program recognizes and uses existing components instead of running commands to create them again because the program meets the dependencies.

Figure 1.1. OpenShift Container Platform installation targets and dependencies



1.1.2. About Red Hat Enterprise Linux CoreOS (RHCOS)

Post-installation, each cluster machine uses Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. RHCOS is the immutable container host version of Red Hat Enterprise Linux (RHEL) and features a RHEL kernel with SELinux enabled by default. RHCOS includes the **kubelet**, which is the Kubernetes node agent, and the CRI-O container runtime, which is optimized for Kubernetes.

Every control plane machine in an OpenShift Container Platform 4.12 cluster must use RHCOS, which includes a critical first-boot provisioning tool called Ignition. This tool enables the cluster to configure the machines. Operating system updates are delivered as a bootable container image, using **OSTree** as a backend, that is deployed across the cluster by the Machine Config Operator. Actual operating system changes are made in-place on each machine as an atomic operation by using **rpm-ostree**. Together, these technologies enable OpenShift Container Platform to manage the operating system like it manages any other application on the cluster, by in-place upgrades that keep the entire platform up to date. These in-place updates can reduce the burden on operations teams.

If you use RHCOS as the operating system for all cluster machines, the cluster manages all aspects of its components and machines, including the operating system. Because of this, only the installation program and the Machine Config Operator can change machines. The installation program uses Ignition config files to set the exact state of each machine, and the Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

1.1.3. Glossary of common terms for OpenShift Container Platform installing

The glossary defines common terms that relate to the installation content. Read the following list of terms to better understand the installation process.

Assisted Installer

An installer hosted at console.redhat.com that provides a web-based user interface or a RESTful API for creating a cluster configuration. The [Assisted Installer](#) generates a discovery image. Cluster machines boot with the discovery image, which installs RHCOS and an agent. Together, the Assisted Installer and agent provide preinstallation validation and installation for the cluster.

Agent-based Installer

An installer similar to the Assisted Installer, but you must download the [Agent-based Installer](#) first. The Agent-based Installer is ideal for disconnected environments.

Bootstrap node

A temporary machine that runs a minimal Kubernetes configuration required to deploy the OpenShift Container Platform control plane.

Control plane

A container orchestration layer that exposes the API and interfaces to define, deploy, and manage the lifecycle of containers. Also known as control plane machines.

Compute node

Nodes that are responsible for executing workloads for cluster users. Also known as worker nodes.

Disconnected installation

In some situations, parts of a data center might not have access to the internet, even through proxy servers. You can still install the OpenShift Container Platform in these environments, but you must download the required software and images and make them available to the disconnected environment.

The OpenShift Container Platform installation program

A program that provisions the infrastructure and deploys a cluster.

Installer-provisioned infrastructure

The installation program deploys and configures the infrastructure that the cluster runs on.

Ignition config files

A file that the Ignition tool uses to configure Red Hat Enterprise Linux CoreOS (RHCOS) during operating system initialization. The installation program generates different Ignition configuration files to initialize bootstrap, control plane, and worker nodes.

Kubernetes manifests

Specifications of a Kubernetes API object in a JSON or YAML format. A configuration file can include deployments, config maps, secrets, daemonsets, and so on.

Kubelet

A primary node agent that runs on each node in the cluster to ensure that containers are running in a pod.

Load balancers

A load balancer serves as the single point of contact for clients. Load balancers for the API distribute incoming traffic across control plane nodes.

Machine Config Operator

An Operator that manages and applies configurations and updates of the base operating system and container runtime, including everything between the kernel and kubelet, for the nodes in the cluster.

Operators

The preferred method of packaging, deploying, and managing a Kubernetes application in an OpenShift Container Platform cluster. An operator takes human operational knowledge and encodes it into software that is easily packaged and shared with customers.

User-provisioned infrastructure

You can install OpenShift Container Platform on infrastructure that you provide. You can use the installation program to generate the assets required to provision the cluster infrastructure, create the cluster infrastructure, and then deploy the cluster to the infrastructure that you provided.

1.1.4. Installation process

Except for the Assisted Installer, when you install an OpenShift Container Platform cluster, you must download the installation program from the appropriate [Cluster Type](#) page on the OpenShift Cluster Manager Hybrid Cloud Console. This console manages:

- REST API for accounts.
- Registry tokens, which are the pull secrets that you use to obtain the required components.
- Cluster registration, which associates the cluster identity to your Red Hat account to facilitate the gathering of usage metrics.

In OpenShift Container Platform 4.12, the installation program is a Go binary file that performs a series of file transformations on a set of assets. The way you interact with the installation program differs depending on your installation type. Consider the following installation use cases:

- To deploy a cluster with the Assisted Installer, you must configure the cluster settings by using the [Assisted Installer](#). There is no installation program to download and configure. After you finish setting the cluster configuration, you download a discovery ISO and then boot cluster machines with that image. You can install clusters with the Assisted Installer on Nutanix, vSphere, and bare metal with full integration, and other platforms without integration. If you install on bare metal, you must provide all of the cluster infrastructure and resources, including the networking, load balancing, storage, and individual cluster machines.
- To deploy clusters with the Agent-based Installer, you can download the [Agent-based Installer](#) first. You can then configure the cluster and generate a discovery image. You boot cluster machines with the discovery image, which installs an agent that communicates with the installation program and handles the provisioning for you instead of you interacting with the installation program or setting up a provisioner machine yourself. You must provide all of the cluster infrastructure and resources, including the networking, load balancing, storage, and individual cluster machines. This approach is ideal for disconnected environments.
- For clusters with installer-provisioned infrastructure, you delegate the infrastructure bootstrapping and provisioning to the installation program instead of doing it yourself. The installation program creates all of the networking, machines, and operating systems that are required to support the cluster, except if you install on bare metal. If you install on bare metal, you must provide all of the cluster infrastructure and resources, including the bootstrap machine, networking, load balancing, storage, and individual cluster machines.
- If you provision and manage the infrastructure for your cluster, you must provide all of the cluster infrastructure and resources, including the bootstrap machine, networking, load balancing, storage, and individual cluster machines.

For the installation program, the program uses three sets of files during installation: an installation configuration file that is named **install-config.yaml**, Kubernetes manifests, and Ignition config files for your machine types.



IMPORTANT

You can modify Kubernetes and the Ignition config files that control the underlying RHCOS operating system during installation. However, no validation is available to confirm the suitability of any modifications that you make to these objects. If you modify these objects, you might render your cluster non-functional. Because of this risk, modifying Kubernetes and Ignition config files is not supported unless you are following documented procedures or are instructed to do so by Red Hat support.

The installation configuration file is transformed into Kubernetes manifests, and then the manifests are wrapped into Ignition config files. The installation program uses these Ignition config files to create the cluster.

The installation configuration files are all pruned when you run the installation program, so be sure to back up all the configuration files that you want to use again.



IMPORTANT

You cannot modify the parameters that you set during installation, but you can modify many cluster attributes after installation.

The installation process with the Assisted Installer

Installation with the [Assisted Installer](#) involves creating a cluster configuration interactively by using the web-based user interface or the RESTful API. The Assisted Installer user interface prompts you for required values and provides reasonable default values for the remaining parameters, unless you change them in the user interface or with the API. The Assisted Installer generates a discovery image, which you download and use to boot the cluster machines. The image installs RHCOS and an agent, and the agent handles the provisioning for you. You can install OpenShift Container Platform with the Assisted Installer and full integration on Nutanix, vSphere, and bare metal. Additionally, you can install OpenShift Container Platform with the Assisted Installer on other platforms without integration.

OpenShift Container Platform manages all aspects of the cluster, including the operating system itself. Each machine boots with a configuration that references resources hosted in the cluster that it joins. This configuration allows the cluster to manage itself as updates are applied.

If possible, use the Assisted Installer feature to avoid having to download and configure the Agent-based Installer.

The installation process with Agent-based infrastructure

Agent-based installation is similar to using the Assisted Installer, except that you must initially download and install the [Agent-based Installer](#). An Agent-based installation is useful when you want the convenience of the Assisted Installer, but you need to install a cluster in a disconnected environment.

If possible, use the Agent-based installation feature to avoid having to create a provisioner machine with a bootstrap VM, and then provision and maintain the cluster infrastructure.

The installation process with installer-provisioned infrastructure

The default installation type uses installer-provisioned infrastructure. By default, the installation program acts as an installation wizard, prompting you for values that it cannot determine on its own and providing reasonable default values for the remaining parameters. You can also customize the installation process to support advanced infrastructure scenarios. The installation program provisions the underlying infrastructure for the cluster.

You can install either a standard cluster or a customized cluster. With a standard cluster, you provide minimum details that are required to install the cluster. With a customized cluster, you can specify more details about the platform, such as the number of machines that the control plane uses, the type of

virtual machine that the cluster deploys, or the CIDR range for the Kubernetes service network.

If possible, use this feature to avoid having to provision and maintain the cluster infrastructure. In all other environments, you use the installation program to generate the assets that you require to provision your cluster infrastructure.

With installer-provisioned infrastructure clusters, OpenShift Container Platform manages all aspects of the cluster, including the operating system itself. Each machine boots with a configuration that references resources hosted in the cluster that it joins. This configuration allows the cluster to manage itself as updates are applied.

The installation process with user-provisioned infrastructure

You can also install OpenShift Container Platform on infrastructure that you provide. You use the installation program to generate the assets that you require to provision the cluster infrastructure, create the cluster infrastructure, and then deploy the cluster to the infrastructure that you provided.

If you do not use infrastructure that the installation program provisioned, you must manage and maintain the cluster resources yourself. The following list details some of these self-managed resources:

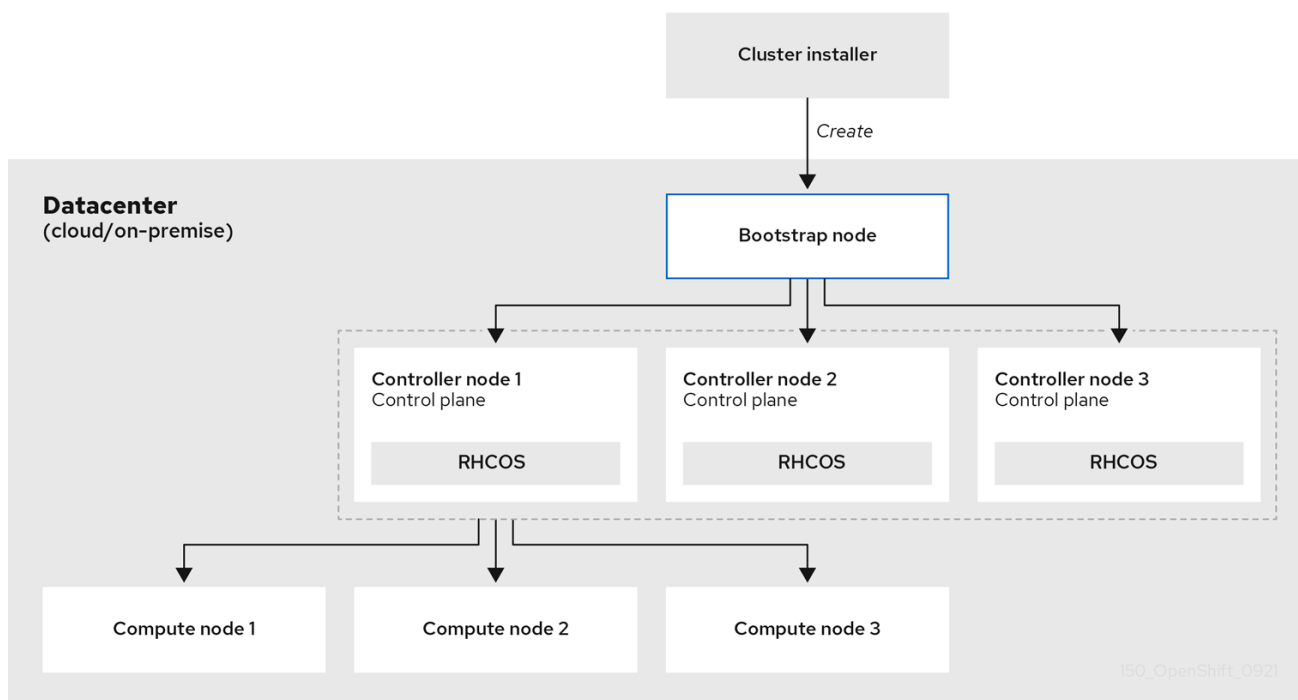
- The underlying infrastructure for the control plane and compute machines that make up the cluster
- Load balancers
- Cluster networking, including the DNS records and required subnets
- Storage for the cluster infrastructure and applications

If your cluster uses user-provisioned infrastructure, you have the option of adding RHEL compute machines to your cluster.

Installation process details

When a cluster is provisioned, each machine in the cluster requires information about the cluster. OpenShift Container Platform uses a temporary bootstrap machine during initial configuration to provide the required information to the permanent control plane. The temporary bootstrap machine boots by using an Ignition config file that describes how to create the cluster. The bootstrap machine creates the control plane machines that make up the control plane. The control plane machines then create the compute machines, which are also known as worker machines. The following figure illustrates this process:

Figure 1.2. Creating the bootstrap, control plane, and compute machines



After the cluster machines initialize, the bootstrap machine is destroyed. All clusters use the bootstrap process to initialize the cluster, but if you provision the infrastructure for your cluster, you must complete many of the steps manually.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- Consider using Ignition config files within 12 hours after they are generated, because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Bootstrapping a cluster involves the following steps:

1. The bootstrap machine boots and starts hosting the remote resources required for the control plane machines to boot. If you provision the infrastructure, this step requires manual intervention.
2. The bootstrap machine starts a single-node etcd cluster and a temporary Kubernetes control plane.
3. The control plane machines fetch the remote resources from the bootstrap machine and finish booting. If you provision the infrastructure, this step requires manual intervention.

4. The temporary control plane schedules the production control plane to the production control plane machines.
5. The Cluster Version Operator (CVO) comes online and installs the etcd Operator. The etcd Operator scales up etcd on all control plane nodes.
6. The temporary control plane shuts down and passes control to the production control plane.
7. The bootstrap machine injects OpenShift Container Platform components into the production control plane.
8. The installation program shuts down the bootstrap machine. If you provision the infrastructure, this step requires manual intervention.
9. The control plane sets up the compute nodes.
10. The control plane installs additional services in the form of a set of Operators.

The result of this bootstrapping process is a running OpenShift Container Platform cluster. The cluster then downloads and configures remaining components needed for the day-to-day operations, including the creation of compute machines in supported environments.

1.1.5. Verifying node state after installation

The OpenShift Container Platform installation completes when the following installation health checks are successful:

- The provisioner can access the OpenShift Container Platform web console.
- All control plane nodes are ready.
- All cluster Operators are available.



NOTE

After the installation completes, the specific cluster Operators responsible for the worker nodes continuously attempt to provision all worker nodes. Some time is required before all worker nodes report as **READY**. For installations on bare metal, wait a minimum of 60 minutes before troubleshooting a worker node. For installations on all other platforms, wait a minimum of 40 minutes before troubleshooting a worker node. A **DEGRADED** state for the cluster Operators responsible for the worker nodes depends on the Operators' own resources and not on the state of the nodes.

After your installation completes, you can continue to monitor the condition of the nodes in your cluster.

Prerequisites

- The installation program resolves successfully in the terminal.

Procedure

1. Show the status of all worker nodes:

```
$ oc get nodes
```

Example output

```

NAME                                STATUS ROLES  AGE  VERSION
example-compute1.example.com        Ready  worker  13m  v1.21.6+bb8d50a
example-compute2.example.com        Ready  worker  13m  v1.21.6+bb8d50a
example-compute4.example.com        Ready  worker  14m  v1.21.6+bb8d50a
example-control1.example.com        Ready  master  52m  v1.21.6+bb8d50a
example-control2.example.com        Ready  master  55m  v1.21.6+bb8d50a
example-control3.example.com        Ready  master  55m  v1.21.6+bb8d50a

```

2. Show the phase of all worker machine nodes:

```
$ oc get machines -A
```

Example output

```

NAMESPACE      NAME                                PHASE    TYPE  REGION  ZONE  AGE
openshift-machine-api  example-zbbt6-master-0            Running              95m
openshift-machine-api  example-zbbt6-master-1            Running              95m
openshift-machine-api  example-zbbt6-master-2            Running              95m
openshift-machine-api  example-zbbt6-worker-0-25bhp      Running              49m
openshift-machine-api  example-zbbt6-worker-0-8b4c2      Running              49m
openshift-machine-api  example-zbbt6-worker-0-jkbqt      Running              49m
openshift-machine-api  example-zbbt6-worker-0-qrl5b      Running              49m

```

Additional resources

- [Getting the BareMetalHost resource](#)
- [Following the installation](#)
- [Validating an installation](#)
- [Agent-based Installer](#)
- [Assisted Installer for OpenShift Container Platform](#)

Installation scope

The scope of the OpenShift Container Platform installation program is intentionally narrow. It is designed for simplicity and ensured success. You can complete many more configuration tasks after installation completes.

Additional resources

- See [Available cluster customizations](#) for details about OpenShift Container Platform configuration resources.

1.1.6. OpenShift Local overview

OpenShift Local supports rapid application development to get started building OpenShift Container Platform clusters. OpenShift Local is designed to run on a local computer to simplify setup and testing, and to emulate the cloud development environment locally with all of the tools needed to develop container-based applications.

Regardless of the programming language you use, OpenShift Local hosts your application and brings a minimal, preconfigured Red Hat OpenShift Container Platform cluster to your local PC without the need for a server-based infrastructure.

On a hosted environment, OpenShift Local can create microservices, convert them into images, and run them in Kubernetes-hosted containers directly on your laptop or desktop running Linux, macOS, or Windows 10 or later.

For more information about OpenShift Local, see [Red Hat OpenShift Local Overview](#).

1.2. SUPPORTED PLATFORMS FOR OPENSIFT CONTAINER PLATFORM CLUSTERS

In OpenShift Container Platform 4.12, you can install a cluster that uses installer-provisioned infrastructure on the following platforms:

- Alibaba Cloud
- Amazon Web Services (AWS)
- Bare metal
- Google Cloud Platform (GCP)
- IBM Cloud® VPC
- Microsoft Azure
- Microsoft Azure Stack Hub
- Nutanix
- Red Hat OpenStack Platform (RHOSP)
 - The latest OpenShift Container Platform release supports both the latest RHOSP long-life release and intermediate release. For complete RHOSP release compatibility, see the [OpenShift Container Platform on RHOSP support matrix](#).
- VMware Cloud (VMC) on AWS
- VMware vSphere

For these clusters, all machines, including the computer that you run the installation process on, must have direct internet access to pull images for platform containers and provide telemetry data to Red Hat.



IMPORTANT

After installation, the following changes are not supported:

- Mixing cloud provider platforms.
- Mixing cloud provider components. For example, using a persistent storage framework from a another platform on the platform where you installed the cluster.

In OpenShift Container Platform 4.12, you can install a cluster that uses user-provisioned infrastructure on the following platforms:

- AWS
- Azure
- Azure Stack Hub
- Bare metal
- GCP
- IBM Power
- IBM Z or IBM® LinuxONE
- RHOSP
 - The latest OpenShift Container Platform release supports both the latest RHOSP long-life release and intermediate release. For complete RHOSP release compatibility, see the [OpenShift Container Platform on RHOSP support matrix](#).
- VMware Cloud on AWS
- VMware vSphere

Depending on the supported cases for the platform, you can perform installations on user-provisioned infrastructure, so that you can run machines with full internet access, place your cluster behind a proxy, or perform a disconnected installation.

In a disconnected installation, you can download the images that are required to install a cluster, place them in a mirror registry, and use that data to install your cluster. While you require internet access to pull images for platform containers, with a disconnected installation on vSphere or bare metal infrastructure, your cluster machines do not require direct internet access.

The [OpenShift Container Platform 4.x Tested Integrations](#) page contains details about integration testing for different platforms.

Additional resources

- See [Supported installation methods for different platforms](#) for more information about the types of installations that are available for each supported platform.
- See [Selecting a cluster installation method and preparing it for users](#) for information about choosing an installation method and preparing the required resources.

CHAPTER 2. SELECTING A CLUSTER INSTALLATION METHOD AND PREPARING IT FOR USERS

Before you install OpenShift Container Platform, decide what kind of installation process to follow and verify that you have all of the required resources to prepare the cluster for users.

2.1. SELECTING A CLUSTER INSTALLATION TYPE

Before you install an OpenShift Container Platform cluster, you need to select the best installation instructions to follow. Think about your answers to the following questions to select the best option.

2.1.1. Do you want to install and manage an OpenShift Container Platform cluster yourself?

If you want to install and manage OpenShift Container Platform yourself, you can install it on the following platforms:

- Alibaba Cloud
- Amazon Web Services (AWS) on 64-bit x86 instances
- Amazon Web Services (AWS) on 64-bit ARM instances
- Microsoft Azure on 64-bit x86 instances
- Microsoft Azure on 64-bit ARM instances
- Microsoft Azure Stack Hub
- Google Cloud Platform (GCP)
- Red Hat OpenStack Platform (RHOSP)
- Red Hat Virtualization (RHV)
- IBM Cloud VPC
- IBM Z or IBM® LinuxONE
- IBM Z or IBM® LinuxONE for Red Hat Enterprise Linux (RHEL) KVM
- IBM Power
- Nutanix
- VMware vSphere
- VMware Cloud (VMC) on AWS
- Bare metal or other platform agnostic infrastructure

You can deploy an OpenShift Container Platform 4 cluster to both on-premise hardware and to cloud hosting services, but all of the machines in a cluster must be in the same data center or cloud hosting service.

If you want to use OpenShift Container Platform but do not want to manage the cluster yourself, you have several managed service options. If you want a cluster that is fully managed by Red Hat, you can use [OpenShift Dedicated](#) or [OpenShift Online](#). You can also use OpenShift as a managed service on Azure, AWS, IBM Cloud VPC, or Google Cloud. For more information about managed services, see the [OpenShift Products](#) page. If you install an OpenShift Container Platform cluster with a cloud virtual machine as a virtual bare metal, the corresponding cloud-based storage is not supported.

2.1.2. Have you used OpenShift Container Platform 3 and want to use OpenShift Container Platform 4?

If you used OpenShift Container Platform 3 and want to try OpenShift Container Platform 4, you need to understand how different OpenShift Container Platform 4 is. OpenShift Container Platform 4 weaves the Operators that package, deploy, and manage Kubernetes applications and the operating system that the platform runs on, Red Hat Enterprise Linux CoreOS (RHCOS), together seamlessly. Instead of deploying machines and configuring their operating systems so that you can install OpenShift Container Platform on them, the RHCOS operating system is an integral part of the OpenShift Container Platform cluster. Deploying the operating system for the cluster machines is part of the installation process for OpenShift Container Platform. See [Differences between OpenShift Container Platform 3 and 4](#).

Because you need to provision machines as part of the OpenShift Container Platform cluster installation process, you cannot upgrade an OpenShift Container Platform 3 cluster to OpenShift Container Platform 4. Instead, you must create a new OpenShift Container Platform 4 cluster and migrate your OpenShift Container Platform 3 workloads to them. For more information about migrating, see [Migrating from OpenShift Container Platform 3 to 4 overview](#). Because you must migrate to OpenShift Container Platform 4, you can use any type of production cluster installation process to create your new cluster.

2.1.3. Do you want to use existing components in your cluster?

Because the operating system is integral to OpenShift Container Platform, it is easier to let the installation program for OpenShift Container Platform stand up all of the infrastructure. These are called *installer provisioned infrastructure* installations. In this type of installation, you can provide some existing infrastructure to the cluster, but the installation program deploys all of the machines that your cluster initially needs.

You can deploy an installer-provisioned infrastructure cluster without specifying any customizations to the cluster or its underlying machines to [Alibaba Cloud](#), [AWS](#), [Azure](#), [Azure Stack Hub](#), [GCP](#), [Nutanix](#), or [VMC on AWS](#). These installation methods are the fastest way to deploy a production-capable OpenShift Container Platform cluster.

If you need to perform basic configuration for your installer-provisioned infrastructure cluster, such as the instance type for the cluster machines, you can customize an installation for [Alibaba Cloud](#), [AWS](#), [Azure](#), [GCP](#), [Nutanix](#), or [VMC on AWS](#).

For installer-provisioned infrastructure installations, you can use an existing [VPC in AWS](#), [vNet in Azure](#), or [VPC in GCP](#). You can also reuse part of your networking infrastructure so that your cluster in [AWS](#), [Azure](#), [GCP](#), or [VMC on AWS](#) can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. If you have existing accounts and credentials on these clouds, you can re-use them, but you might need to modify the accounts to have the required permissions to install OpenShift Container Platform clusters on them.

You can use the installer-provisioned infrastructure method to create appropriate machine instances on your hardware for [RHOSP](#), [RHOSP with Kuryr](#), [RHV](#), [vSphere](#), and [bare metal](#). Additionally, for [vSphere](#), [VMC on AWS](#), you can also customize additional network parameters during installation.

If you want to reuse extensive cloud infrastructure, you can complete a *user-provisioned infrastructure*

installation. With these installations, you manually deploy the machines that your cluster requires during the installation process. If you perform a user-provisioned infrastructure installation on [AWS](#), [Azure](#), [Azure Stack Hub](#), [GCP](#), or [VMC on AWS](#), you can use the provided templates to help you stand up all of the required components. You can also reuse a shared [VPC on GCP](#). Otherwise, you can use the [provider-agnostic](#) installation method to deploy a cluster into other clouds.

You can also complete a user-provisioned infrastructure installation on your existing hardware. If you use [RHOSP](#), [RHV](#), [IBM Z or IBM® LinuxONE](#), [IBM Z and IBM® LinuxONE with RHEL KVM](#), [IBM Power](#), or [vSphere](#), use the specific installation instructions to deploy your cluster. If you use other supported hardware, follow the [bare metal installation](#) procedure. For some of these platforms, such as [RHOSP](#), [vSphere](#), [VMC on AWS](#), and [bare metal](#), you can also customize additional network parameters during installation.

2.1.4. Do you need extra security for your cluster?

If you use a user-provisioned installation method, you can configure a proxy for your cluster. The instructions are included in each installation procedure.

If you want to prevent your cluster on a public cloud from exposing endpoints externally, you can deploy a private cluster with installer-provisioned infrastructure on [AWS](#), [Azure](#), or [GCP](#).

If you need to install your cluster that has limited access to the internet, such as a disconnected or restricted network cluster, you can [mirror the installation packages](#) and install the cluster from them. Follow detailed instructions for user provisioned infrastructure installations into restricted networks for [AWS](#), [GCP](#), [IBM Z or IBM® LinuxONE](#), [IBM Z or IBM® LinuxONE with RHEL KVM](#), [IBM Power](#), [vSphere](#), [VMC on AWS](#), or [bare metal](#). You can also install a cluster into a restricted network using installer-provisioned infrastructure by following detailed instructions for [AWS](#), [GCP](#), [Nutanix](#), [VMC on AWS](#), [RHOSP](#), [RHV](#), and [vSphere](#).

If you need to deploy your cluster to an [AWS GovCloud region](#), [AWS China region](#), or [Azure government region](#), you can configure those custom regions during an installer-provisioned infrastructure installation.

You can also configure the cluster machines to use [FIPS Validated / Modules in Process cryptographic libraries](#) during installation.



IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

2.2. PREPARING YOUR CLUSTER FOR USERS AFTER INSTALLATION

Some configuration is not required to install the cluster but recommended before your users access the cluster. You can customize the cluster itself by [customizing](#) the Operators that make up your cluster and integrate you cluster with other required systems, such as an identity provider.

For a production cluster, you must configure the following integrations:

- [Persistent storage](#)
- [An identity provider](#)
- [Monitoring core OpenShift Container Platform components](#)

2.3. PREPARING YOUR CLUSTER FOR WORKLOADS

Depending on your workload needs, you might need to take extra steps before you begin deploying applications. For example, after you prepare infrastructure to support your application [build strategy](#), you might need to make provisions for [low-latency](#) workloads or to [protect sensitive workloads](#). You can also configure [monitoring](#) for application workloads. If you plan to run [Windows workloads](#), you must enable [hybrid networking with OVN-Kubernetes](#) during the installation process; hybrid networking cannot be enabled after your cluster is installed.

2.4. SUPPORTED INSTALLATION METHODS FOR DIFFERENT PLATFORMS

You can perform different types of installations on different platforms.



NOTE

Not all installation options are supported for all platforms, as shown in the following tables. A checkmark indicates that the option is supported and links to the relevant section.

Table 2.1. Installer-provisioned infrastructure options

	Alibaba Cloud	Amazon Web Services	Azure	Azure Stack	Google Cloud Platform	IBM Cloud	IBM Z	IBM Power	Red Hat OpenShift	Red Hat OpenShift VM	Bare Metal	Bare Metal ARM	vSphere	VMware Tanzu	IBM Cloud	IBM Z	IBM Power
Default	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		
Custom	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓		

	Alibaba	AWS	AWS	Azure	Azure	Azure	Google	Netflix	RHO	RHV	Baremetal	Baremetal	vSphere	VMC	IBM	IBM	IBM	
		(64-bit x86)	(64-bit ARM)	(64-bit x86)	(64-bit ARM)	(64-bit ARM)	Cloud	Cloud	OSP		(64-bit x86)	(64-bit ARM)			Cloud	Z	Power	
Network customization	✓	✓	✓	✓	✓	✓	✓		✓				✓	✓	✓			
Restricted network		✓	✓				✓	✓	✓	✓	✓	✓	✓	✓				
Private clusters		✓	✓	✓	✓		✓								✓			

	Alibaba	AWS (64-bit x86_64)	AWS (64-bit ARM)	Azure (64-bit x86_64)	Azure (64-bit ARM)	Azure Stack Hub	Google Cloud Platform	Nutanix	RHOCP	RHV	Baremetal (64-bit x86_64)	Baremetal (64-bit ARM)	vSphere	VMC	IBM Cloud VPC	IBM Z	IBM Power
Existing virtual private networks		✓	✓	✓	✓		✓									✓	
Government regions	✓			✓													
Secret regions	✓																

Alibaba	AWSS (64-bit x86)	AWSS (64-bit ARM)	Azure (64-bit x86)	Azure (64-bit ARM)	Azure Stack Hub	Google Cloud	Nutanix	RHOSSP	RHOSSP	Baremetal (64-bit x86)	Baremetal (64-bit ARM)	vSphere	VMC	IBM Cloud VPC	IBM Z	IBM Power
China regions	✓															

Table 2.2. User-provisioned infrastructure options

Alibaba	AWSS (64-bit x86)	AWSS (64-bit ARM)	Azure	Azure Stack Hub	Google Cloud	Nutanix	RHOSSP	RHOSSP	Baremetal (64-bit x86)	Baremetal (64-bit ARM)	vSphere	VMC	IBM Cloud VPC	IBM Z with RHEL KVM	IBM Power	Platform agnostic
Custom	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓

	Al ib a b a	A W S (6 4- bit x8 6)	A W S (6 4- bit A R M)	A zu re	A zu re St ac k H u b	G C P	N ut an ix	R H O S P	R H V	B ar e m et al (6 4- bit x8 6)	B ar e m et al (6 4- bit A R M)	vS p h er e	V M C	IB M Cl o u d V P C	IB M Z	IB M Z wi th R E L K V M	IB M P o w er	Pl at fo r m a g n o st ic
Net w o r k c u s t o m i z a t i o n								✓		✓	✓	✓	✓					
R e s t r i c t e d n e t w o r k		✓	✓			✓			✓	✓	✓	✓	✓		✓	✓	✓	

Al ib a b a	A W S (6 4- bit x8 6)	A W S (6 4- bit A R M)	A zu re	A zu re St ac k H u b	G C P	N ut an ix	R H O S P	R H V	B ar e m et al (6 4- bit x8 6)	B ar e m et al (6 4- bit A R M)	vS p h er e	V M C	IB M Cl o u d V P C	IB M Z	IB M Z wi th R E L K V M	IB M P o w er	Pl at fo r m a g n o st ic
S h ar e d V P C h os te d o ut si de of cl us te r pr oj e ct					✓												

CHAPTER 3. CLUSTER CAPABILITIES

Cluster administrators can use cluster capabilities to enable or disable optional components prior to installation. Cluster administrators can enable cluster capabilities at anytime after installation.



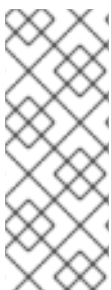
NOTE

Cluster administrators cannot disable a cluster capability after it is enabled.

3.1. SELECTING CLUSTER CAPABILITIES

You can select cluster capabilities by following one of the installation methods that include customizing your cluster, such as "Installing a cluster on AWS with customizations" or "Installing a cluster on GCP with customizations".

During a customized installation, you create an **install-config.yaml** file that contains the configuration parameters for your cluster.



NOTE

If you customize your cluster by enabling or disabling specific cluster capabilities, you are responsible for manually maintaining your **install-config.yaml** file. New OpenShift Container Platform updates might declare new capability handles for existing components, or introduce new components altogether. Users who customize their **install-config.yaml** file should consider periodically updating their **install-config.yaml** file as OpenShift Container Platform is updated.

You can use the following configuration parameters to select cluster capabilities:

capabilities:

```
baselineCapabilitySet: v4.11 1
additionalEnabledCapabilities: 2
- CSISnapshot
- Console
- Storage
```

- 1** Defines a baseline set of capabilities to install. Valid values are **None**, **vCurrent** and **v4.x**. If you select **None**, all optional capabilities will be disabled. The default value is **vCurrent**, which enables all optional capabilities.



NOTE

v4.x refers to any value up to and including the current cluster version. For example, valid values for a OpenShift Container Platform 4.12 cluster are **v4.11** and **v4.12**.

- 2** Defines a list of capabilities to explicitly enable. These will be enabled in addition to the capabilities specified in **baselineCapabilitySet**.

**NOTE**

In this example, the default capability is set to **v4.11**. The **additionalEnabledCapabilities** field enables additional capabilities over the default **v4.11** capability set.

The following table describes the **baselineCapabilitySet** values.

Table 3.1. Cluster capabilities **baselineCapabilitySet values description**

Value	Description
vCurrent	Specify this option when you want to automatically add new, default capabilities that are introduced in new releases.
v4.11	Specify this option when you want to enable the default capabilities for OpenShift Container Platform 4.11. By specifying v4.11 , capabilities that are introduced in newer versions of OpenShift Container Platform are not enabled. The default capabilities in OpenShift Container Platform 4.11 are baremetal, marketplace, and openshift-samples .
v4.12	Specify this option when you want to enable the default capabilities for OpenShift Container Platform 4.12. By specifying v4.12 , capabilities that are introduced in newer versions of OpenShift Container Platform are not enabled. The default capabilities in OpenShift Container Platform 4.12 are baremetal, marketplace, openshift-samples, Console, Insights, Storage and CSISnapshot .
None	Specify when the other sets are too large, and you do not need any capabilities or want to fine-tune via additionalEnabledCapabilities .

Additional resources

- [Installing a cluster on AWS with customizations](#)
- [Installing a cluster on GCP with customizations](#)

3.2. OPTIONAL CLUSTER CAPABILITIES IN OPENSIFT CONTAINER PLATFORM 4.12

Currently, cluster Operators provide the features for these optional capabilities. The following summarizes the features provided by each capability and what functionality you lose if it is disabled.

Additional resources

- [Cluster Operators reference](#)

3.2.1. Bare-metal capability

Purpose

The Cluster Baremetal Operator provides the features for the **baremetal** capability.

The Cluster Baremetal Operator (CBO) deploys all the components necessary to take a bare-metal server to a fully functioning worker node ready to run OpenShift Container Platform compute nodes. The CBO ensures that the metal3 deployment, which consists of the Bare Metal Operator (BMO) and Ironic containers, runs on one of the control plane nodes within the OpenShift Container Platform cluster. The CBO also listens for OpenShift Container Platform updates to resources that it watches and takes appropriate action.

The bare-metal capability is required for deployments using installer-provisioned infrastructure. Disabling the bare-metal capability can result in unexpected problems with these deployments.

It is recommended that cluster administrators only disable the bare-metal capability during installations with user-provisioned infrastructure that do not have any **BareMetalHost** resources in the cluster.



IMPORTANT

If the bare-metal capability is disabled, the cluster cannot provision or manage bare-metal nodes. Only disable the capability if there are no **BareMetalHost** resources in your deployment.

Additional resources

- [Deploying installer-provisioned clusters on bare metal](#)
- [Preparing for bare metal cluster installation](#)
- [Bare metal configuration](#)

3.2.2. Cluster storage capability

Purpose

The Cluster Storage Operator provides the features for the **Storage** capability.

The Cluster Storage Operator sets OpenShift Container Platform cluster-wide storage defaults. It ensures a default **storageclass** exists for OpenShift Container Platform clusters. It also installs Container Storage Interface (CSI) drivers which enable your cluster to use various storage backends.



IMPORTANT

If the cluster storage capability is disabled, the cluster will not have a default **storageclass** or any CSI drivers. Users with administrator privileges can create a default **storageclass** and manually install CSI drivers if the cluster storage capability is disabled.

Notes

- The storage class that the Operator creates can be made non-default by editing its annotation, but this storage class cannot be deleted as long as the Operator runs.

3.2.3. Console capability

Purpose

The Console Operator provides the features for the **Console** capability.

The Console Operator installs and maintains the OpenShift Container Platform web console on a cluster. The Console Operator is installed by default and automatically maintains a console.

Additional resources

- [Web console overview](#)

3.2.4. CSI snapshot controller capability

Purpose

The Cluster CSI Snapshot Controller Operator provides the features for the **CSISnapshot** capability.

The Cluster CSI Snapshot Controller Operator installs and maintains the CSI Snapshot Controller. The CSI Snapshot Controller is responsible for watching the **VolumeSnapshot** CRD objects and manages the creation and deletion lifecycle of volume snapshots.

Additional resources

- [CSI volume snapshots](#)

3.2.5. Insights capability

Purpose

The Insights Operator provides the features for the **Insights** capability.

The Insights Operator gathers OpenShift Container Platform configuration data and sends it to Red Hat. The data is used to produce proactive insights recommendations about potential issues that a cluster might be exposed to. These insights are communicated to cluster administrators through Insights Advisor on console.redhat.com.

Notes

Insights Operator complements OpenShift Container Platform Telemetry.

Additional resources

- [Using Insights Operator](#)

3.2.6. Marketplace capability

Purpose

The Marketplace Operator provides the features for the **marketplace** capability.

The Marketplace Operator simplifies the process for bringing off-cluster Operators to your cluster by using a set of default Operator Lifecycle Manager (OLM) catalogs on the cluster. When the Marketplace Operator is installed, it creates the **openshift-marketplace** namespace. OLM ensures catalog sources installed in the **openshift-marketplace** namespace are available for all namespaces on the cluster.

If you disable the **marketplace** capability, the Marketplace Operator does not create the **openshift-marketplace** namespace. Catalog sources can still be configured and managed on the cluster manually, but OLM depends on the **openshift-marketplace** namespace in order to make catalogs available to all namespaces on the cluster. Users with elevated permissions to create namespaces prefixed with **openshift-**, such as system or cluster administrators, can manually create the **openshift-marketplace** namespace.

If you enable the **marketplace** capability, you can enable and disable individual catalogs by configuring the Marketplace Operator.

Additional resources

- [Red Hat-provided Operator catalogs](#)

3.2.7. OpenShift samples capability

Purpose

The Cluster Samples Operator provides the features for the **openshift-samples** capability.

The Cluster Samples Operator manages the sample image streams and templates stored in the **openshift** namespace.

On initial start up, the Operator creates the default samples configuration resource to initiate the creation of the image streams and templates. The configuration object is a cluster scoped object with the key **cluster** and type **configs.samples**.

The image streams are the Red Hat Enterprise Linux CoreOS (RHCOS)-based OpenShift Container Platform image streams pointing to images on **registry.redhat.io**. Similarly, the templates are those categorized as OpenShift Container Platform templates.

If you disable the samples capability, users cannot access the image streams, samples, and templates it provides. Depending on your deployment, you might want to disable this component if you do not need it.

Additional resources

- [Configuring the Cluster Samples Operator](#)

3.3. ADDITIONAL RESOURCES

- [Enabling cluster capabilities after installation](#)

CHAPTER 4. DISCONNECTED INSTALLATION MIRRORING

4.1. ABOUT DISCONNECTED INSTALLATION MIRRORING

You can use a mirror registry to ensure that your clusters only use container images that satisfy your organizational controls on external content. Before you install a cluster on infrastructure that you provision in a restricted network, you must mirror the required container images into that environment. To mirror container images, you must have a registry for mirroring.

4.1.1. Creating a mirror registry

If you already have a container image registry, such as Red Hat Quay, you can use it as your mirror registry. If you do not already have a registry, you can [create a mirror registry using the *mirror registry for Red Hat OpenShift*](#).

4.1.2. Mirroring images for a disconnected installation

You can use one of the following procedures to mirror your OpenShift Container Platform image repository to your mirror registry:

- [Mirroring images for a disconnected installation](#)
- [Mirroring images for a disconnected installation using the *oc-mirror* plugin](#)

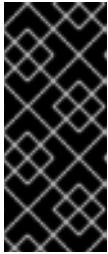
4.2. CREATING A MIRROR REGISTRY WITH MIRROR REGISTRY FOR RED HAT OPENSIFT

The *mirror registry for Red Hat OpenShift* is a small and streamlined container registry that you can use as a target for mirroring the required container images of OpenShift Container Platform for disconnected installations.

If you already have a container image registry, such as Red Hat Quay, you can skip this section and go straight to [Mirroring the OpenShift Container Platform image repository](#).

4.2.1. Prerequisites

- An OpenShift Container Platform subscription.
- Red Hat Enterprise Linux (RHEL) 8 and 9 with Podman 3.4.2 or later and OpenSSL installed.
- Fully qualified domain name for the Red Hat Quay service, which must resolve through a DNS server.
- Key-based SSH connectivity on the target host. SSH keys are automatically generated for local installs. For remote hosts, you must generate your own SSH keys.
- 2 or more vCPUs.
- 8 GB of RAM.
- About 12 GB for OpenShift Container Platform 4.12 release images, or about 358 GB for OpenShift Container Platform 4.12 release images and OpenShift Container Platform 4.12 Red Hat Operator images. Up to 1 TB per stream or more is suggested.



IMPORTANT

These requirements are based on local testing results with only release images and Operator images. Storage requirements can vary based on your organization's needs. You might require more space, for example, when you mirror multiple z-streams. You can use standard [Red Hat Quay functionality](#) or the proper [API callout](#) to remove unnecessary images and free up space.

4.2.2. Mirror registry for Red Hat OpenShift introduction

For disconnected deployments of OpenShift Container Platform, a container registry is required to carry out the installation of the clusters. To run a production-grade registry service on such a cluster, you must create a separate registry deployment to install the first cluster. The *mirror registry for Red Hat OpenShift* addresses this need and is included in every OpenShift subscription. It is available for download on the [OpenShift console Downloads](#) page.

The *mirror registry for Red Hat OpenShift* allows users to install a small-scale version of Red Hat Quay and its required components using the **mirror-registry** command line interface (CLI) tool. The *mirror registry for Red Hat OpenShift* is deployed automatically with preconfigured local storage and a local database. It also includes auto-generated user credentials and access permissions with a single set of inputs and no additional configuration choices to get started.

The *mirror registry for Red Hat OpenShift* provides a pre-determined network configuration and reports deployed component credentials and access URLs upon success. A limited set of optional configuration inputs like fully qualified domain name (FQDN) services, superuser name and password, and custom TLS certificates are also provided. This provides users with a container registry so that they can easily create an offline mirror of all OpenShift Container Platform release content when running OpenShift Container Platform in restricted network environments.

Use of the *mirror registry for Red Hat OpenShift* is optional if another container registry is already available in the install environment.

4.2.2.1. Mirror registry for Red Hat OpenShift limitations

The following limitations apply to the *mirror registry for Red Hat OpenShift*:

- The *mirror registry for Red Hat OpenShift* is not a highly-available registry and only local file system storage is supported. It is not intended to replace Red Hat Quay or the internal image registry for OpenShift Container Platform.
- The *mirror registry for Red Hat OpenShift* is only supported for hosting images that are required to install a disconnected OpenShift Container Platform cluster, such as Release images or Red Hat Operator images. It uses local storage on your Red Hat Enterprise Linux (RHEL) machine, and storage supported by RHEL is supported by the *mirror registry for Red Hat OpenShift*.



NOTE

Because the *mirror registry for Red Hat OpenShift* uses local storage, you should remain aware of the storage usage consumed when mirroring images and use Red Hat Quay's garbage collection feature to mitigate potential issues. For more information about this feature, see "Red Hat Quay garbage collection".

- Support for Red Hat product images that are pushed to the *mirror registry for Red Hat OpenShift* for bootstrapping purposes are covered by valid subscriptions for each respective product. A list of exceptions to further enable the bootstrap experience can be found on the

[Self-managed Red Hat OpenShift sizing and subscription guide](#) .

- Content built by customers should not be hosted by the *mirror registry for Red Hat OpenShift* .
- Using the *mirror registry for Red Hat OpenShift* with more than one cluster is discouraged because multiple clusters can create a single point of failure when updating your cluster fleet. It is advised to leverage the *mirror registry for Red Hat OpenShift* to install a cluster that can host a production-grade, highly-available registry such as Red Hat Quay, which can serve OpenShift Container Platform content to other clusters.

4.2.3. Mirroring on a local host with mirror registry for Red Hat OpenShift

This procedure explains how to install the *mirror registry for Red Hat OpenShift* on a local host using the **mirror-registry** installer tool. By doing so, users can create a local host registry running on port 443 for the purpose of storing a mirror of OpenShift Container Platform images.



NOTE

Installing the *mirror registry for Red Hat OpenShift* using the **mirror-registry** CLI tool makes several changes to your machine. After installation, a **\$HOME/quay-install** directory is created, which has installation files, local storage, and the configuration bundle. Trusted SSH keys are generated in case the deployment target is the local host, and systemd files on the host machine are set up to ensure that container runtimes are persistent. Additionally, an initial user named **init** is created with an automatically generated password. All access credentials are printed at the end of the install routine.

Procedure

1. Download the **mirror-registry.tar.gz** package for the latest version of the *mirror registry for Red Hat OpenShift* found on the [OpenShift console Downloads](#) page.
2. Install the *mirror registry for Red Hat OpenShift* on your local host with your current user account by using the **mirror-registry** tool. For a full list of available flags, see "mirror registry for Red Hat OpenShift flags".

```
$ ./mirror-registry install \
  --quayHostname <host_example_com> \
  --quayRoot <example_directory_name>
```

3. Use the user name and password generated during installation to log into the registry by running the following command:

```
$ podman login -u init \
  -p <password> \
  <host_example_com>:8443> \
  --tls-verify=false 1
```

- 1** You can avoid running **--tls-verify=false** by configuring your system to trust the generated rootCA certificates. See "Using SSL to protect connections to Red Hat Quay" and "Configuring the system to trust the certificate authority" for more information.

**NOTE**

You can also log in by accessing the UI at <https://<host.example.com>:8443> after installation.

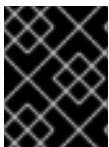
4. You can mirror OpenShift Container Platform images after logging in. Depending on your needs, see either the "Mirroring the OpenShift Container Platform image repository" or the "Mirroring Operator catalogs for use with disconnected clusters" sections of this document.

**NOTE**

If there are issues with images stored by the *mirror registry for Red Hat OpenShift* due to storage layer problems, you can remirror the OpenShift Container Platform images, or reinstall mirror registry on more stable storage.

4.2.4. Updating mirror registry for Red Hat OpenShift from a local host

This procedure explains how to update the *mirror registry for Red Hat OpenShift* from a local host using the **upgrade** command. Updating to the latest version ensures new features, bug fixes, and security vulnerability fixes.

**IMPORTANT**

When updating, there is intermittent downtime of your mirror registry, as it is restarted during the update process.

Prerequisites

- You have installed the *mirror registry for Red Hat OpenShift* on a local host.

Procedure

- If you are upgrading the *mirror registry for Red Hat OpenShift* from 1.2.z → 1.3.0, and your installation directory is the default at **/etc/quay-install**, you can enter the following command:

```
$ sudo ./mirror-registry upgrade -v
```

**NOTE**

- *mirror registry for Red Hat OpenShift* migrates Podman volumes for Quay storage, Postgres data, and **/etc/quay-install** data to the new **\$HOME/quay-install** location. This allows you to use *mirror registry for Red Hat OpenShift* without the **--quayRoot** flag during future upgrades.
- Users who upgrade *mirror registry for Red Hat OpenShift* with the **./mirror-registry upgrade -v** flag must include the same credentials used when creating their mirror registry. For example, if you installed the *mirror registry for Red Hat OpenShift* with **--quayHostname <host_example_com>** and **--quayRoot <example_directory_name>**, you must include that string to properly upgrade the mirror registry.

- If you are upgrading the *mirror registry for Red Hat OpenShift* from 1.2.z → 1.3.0 and you used a specified directory in your 1.2.z deployment, you must pass in the new **--pgStorage** and **--quayStorage** flags. For example:

```
$ sudo ./mirror-registry upgrade --quayHostname <host_example_com> --quayRoot
<example_directory_name> --pgStorage <example_directory_name>/pg-data --quayStorage
<example_directory_name>/quay-storage -v
```

4.2.5. Mirroring on a remote host with mirror registry for Red Hat OpenShift

This procedure explains how to install the *mirror registry for Red Hat OpenShift* on a remote host using the **mirror-registry** tool. By doing so, users can create a registry to hold a mirror of OpenShift Container Platform images.



NOTE

Installing the *mirror registry for Red Hat OpenShift* using the **mirror-registry** CLI tool makes several changes to your machine. After installation, a **\$HOME/quay-install** directory is created, which has installation files, local storage, and the configuration bundle. Trusted SSH keys are generated in case the deployment target is the local host, and systemd files on the host machine are set up to ensure that container runtimes are persistent. Additionally, an initial user named **init** is created with an automatically generated password. All access credentials are printed at the end of the install routine.

Procedure

1. Download the **mirror-registry.tar.gz** package for the latest version of the *mirror registry for Red Hat OpenShift* found on the [OpenShift console Downloads](#) page.
2. Install the *mirror registry for Red Hat OpenShift* on your local host with your current user account by using the **mirror-registry** tool. For a full list of available flags, see "mirror registry for Red Hat OpenShift flags".

```
$ ./mirror-registry install -v \
--targetHostname <host_example_com> \
--targetUsername <example_user> \
-k ~/.ssh/my_ssh_key \
--quayHostname <host_example_com> \
--quayRoot <example_directory_name>
```

3. Use the user name and password generated during installation to log into the mirror registry by running the following command:

```
$ podman login -u init \
-p <password> \
<host_example_com>:8443 \
--tls-verify=false 1
```

- 1** You can avoid running **--tls-verify=false** by configuring your system to trust the generated rootCA certificates. See "Using SSL to protect connections to Red Hat Quay" and "Configuring the system to trust the certificate authority" for more information.

**NOTE**

You can also log in by accessing the UI at <https://<host.example.com>:8443> after installation.

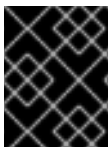
4. You can mirror OpenShift Container Platform images after logging in. Depending on your needs, see either the "Mirroring the OpenShift Container Platform image repository" or the "Mirroring Operator catalogs for use with disconnected clusters" sections of this document.

**NOTE**

If there are issues with images stored by the *mirror registry for Red Hat OpenShift* due to storage layer problems, you can remirror the OpenShift Container Platform images, or reinstall mirror registry on more stable storage.

4.2.6. Updating mirror registry for Red Hat OpenShift from a remote host

This procedure explains how to update the *mirror registry for Red Hat OpenShift* from a remote host using the **upgrade** command. Updating to the latest version ensures bug fixes and security vulnerability fixes.

**IMPORTANT**

When updating, there is intermittent downtime of your mirror registry, as it is restarted during the update process.

Prerequisites

- You have installed the *mirror registry for Red Hat OpenShift* on a remote host.

Procedure

- To upgrade the *mirror registry for Red Hat OpenShift* from a remote host, enter the following command:

```
$ ./mirror-registry upgrade -v --targetHostname <remote_host_url> --targetUsername
<user_name> -k ~/.ssh/my_ssh_key
```

**NOTE**

Users who upgrade the *mirror registry for Red Hat OpenShift* with the **./mirror-registry upgrade -v** flag must include the same credentials used when creating their mirror registry. For example, if you installed the *mirror registry for Red Hat OpenShift* with **--quayHostname <host_example_com>** and **--quayRoot <example_directory_name>**, you must include that string to properly upgrade the mirror registry.

4.2.7. Replacing mirror registry for Red Hat OpenShift SSL/TLS certificates

In some cases, you might want to update your SSL/TLS certificates for the *mirror registry for Red Hat OpenShift*. This is useful in the following scenarios:

- If you are replacing the current *mirror registry for Red Hat OpenShift* certificate.

- If you are using the same certificate as the previous *mirror registry for Red Hat OpenShift* installation.
- If you are periodically updating the *mirror registry for Red Hat OpenShift* certificate.

Use the following procedure to replace *mirror registry for Red Hat OpenShift* SSL/TLS certificates.

Prerequisites

- You have downloaded the `./mirror-registry` binary from the [OpenShift console Downloads](#) page.

Procedure

1. Enter the following command to install the *mirror registry for Red Hat OpenShift*:

```
$ ./mirror-registry install \
--quayHostname <host_example_com> \
--quayRoot <example_directory_name>
```

This installs the *mirror registry for Red Hat OpenShift* to the `$HOME/quay-install` directory.

2. Prepare a new certificate authority (CA) bundle and generate new `ssl.key` and `ssl.crt` key files. For more information, see [Using SSL/TLS to protect connections to Red Hat Quay](#).
3. Assign `/$HOME/quay-install` an environment variable, for example, `QUAY`, by entering the following command:

```
$ export QUAY=/$HOME/quay-install
```

4. Copy the new `ssl.crt` file to the `/$HOME/quay-install` directory by entering the following command:

```
$ cp ~/ssl.crt $QUAY/quay-config
```

5. Copy the new `ssl.key` file to the `/$HOME/quay-install` directory by entering the following command:

```
$ cp ~/ssl.key $QUAY/quay-config
```

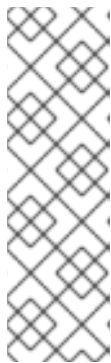
6. Restart the `quay-app` application pod by entering the following command:

```
$ systemctl restart quay-app
```

4.2.8. Uninstalling the mirror registry for Red Hat OpenShift

- You can uninstall the *mirror registry for Red Hat OpenShift* from your local host by running the following command:

```
$ ./mirror-registry uninstall -v \
--quayRoot <example_directory_name>
```



NOTE

- Deleting the *mirror registry for Red Hat OpenShift* will prompt the user before deletion. You can use **--autoApprove** to skip this prompt.
- Users who install the *mirror registry for Red Hat OpenShift* with the **--quayRoot** flag must include the **--quayRoot** flag when uninstalling. For example, if you installed the *mirror registry for Red Hat OpenShift* with **--quayRoot example_directory_name**, you must include that string to properly uninstall the mirror registry.

4.2.9. Mirror registry for Red Hat OpenShift flags

The following flags are available for the *mirror registry for Red Hat OpenShift*:

Flags	Description
--autoApprove	A boolean value that disables interactive prompts. If set to true , the quayRoot directory is automatically deleted when uninstalling the mirror registry. Defaults to false if left unspecified.
--initPassword	The password of the init user created during Quay installation. Must be at least eight characters and contain no whitespace.
--initUser string	Shows the username of the initial user. Defaults to init if left unspecified.
--no-color, -c	Allows users to disable color sequences and propagate that to Ansible when running install, uninstall, and upgrade commands.
--pgStorage	The folder where Postgres persistent storage data is saved. Defaults to the pg-storage Podman volume. Root privileges are required to uninstall.
--quayHostname	The fully-qualified domain name of the mirror registry that clients will use to contact the registry. Equivalent to SERVER_HOSTNAME in the Quay config.yaml . Must resolve by DNS. Defaults to <targetHostname>:8443 if left unspecified. ^[1]
--quayStorage	The folder where Quay persistent storage data is saved. Defaults to the quay-storage Podman volume. Root privileges are required to uninstall.
--quayRoot, -r	The directory where container image layer and configuration data is saved, including rootCA.key , rootCA.pem , and rootCA.srl certificates. Defaults to \$HOME/quay-install if left unspecified.
--ssh-key, -k	The path of your SSH identity key. Defaults to ~/ssh/quay_installer if left unspecified.
--sslCert	The path to the SSL/TLS public key / certificate. Defaults to {quayRoot}/quay-config and is auto-generated if left unspecified.

Flags	Description
--sslCheckSkip	Skips the check for the certificate hostname against the SERVER_HOSTNAME in the config.yaml file. ^[2]
--sslKey	The path to the SSL/TLS private key used for HTTPS communication. Defaults to {quayRoot}/quay-config and is auto-generated if left unspecified.
--targetHostname, -H	The hostname of the target you want to install Quay to. Defaults to \$HOST , for example, a local host, if left unspecified.
--targetUsername, -u	The user on the target host which will be used for SSH. Defaults to \$USER , for example, the current user if left unspecified.
--verbose, -v	Shows debug logs and Ansible playbook outputs.
--version	Shows the version for the <i>mirror registry for Red Hat OpenShift</i>

1. **--quayHostname** must be modified if the public DNS name of your system is different from the local hostname. Additionally, the **--quayHostname** flag does not support installation with an IP address. Installation with a hostname is required.
2. **--sslCheckSkip** is used in cases when the mirror registry is set behind a proxy and the exposed hostname is different from the internal Quay hostname. It can also be used when users do not want the certificates to be validated against the provided Quay hostname during installation.

4.2.10. Mirror registry for Red Hat OpenShift release notes

The *mirror registry for Red Hat OpenShift* is a small and streamlined container registry that you can use as a target for mirroring the required container images of OpenShift Container Platform for disconnected installations.

These release notes track the development of the *mirror registry for Red Hat OpenShift* in OpenShift Container Platform.

For an overview of the *mirror registry for Red Hat OpenShift*, see [Creating a mirror registry with mirror registry for Red Hat OpenShift](#).

4.2.10.1. Mirror registry for Red Hat OpenShift 1.3.11

Issued: 2024-04-23

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.15.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2024:1758 - mirror registry for Red Hat OpenShift 1.3.11](#)

4.2.10.2. Mirror registry for Red Hat OpenShift 1.3.10

Issued: 2023-12-07

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.14.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:7628 - mirror registry for Red Hat OpenShift 1.3.10](#)

4.2.10.3. Mirror registry for Red Hat OpenShift 1.3.9

Issued: 2023-09-19

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.12.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:5241 - mirror registry for Red Hat OpenShift 1.3.9](#)

4.2.10.4. Mirror registry for Red Hat OpenShift 1.3.8

Issued: 2023-08-16

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.11.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:4622 - mirror registry for Red Hat OpenShift 1.3.8](#)

4.2.10.5. Mirror registry for Red Hat OpenShift 1.3.7

Issued: 2023-07-19

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.10.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:4087 - mirror registry for Red Hat OpenShift 1.3.7](#)

4.2.10.6. Mirror registry for Red Hat OpenShift 1.3.6

Issued: 2023-05-30

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.8.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:3302 - mirror registry for Red Hat OpenShift 1.3.6](#)

4.2.10.7. Mirror registry for Red Hat OpenShift 1.3.5

Issued: 2023-05-18

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.7.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:3225 - mirror registry for Red Hat OpenShift 1.3.5](#)

4.2.10.8. Mirror registry for Red Hat OpenShift 1.3.4

Issued: 2023-04-25

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.6.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:1914 - mirror registry for Red Hat OpenShift 1.3.4](#)

4.2.10.9. Mirror registry for Red Hat OpenShift 1.3.3

Issued: 2023-04-05

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.5.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:1528 - mirror registry for Red Hat OpenShift 1.3.3](#)

4.2.10.10. Mirror registry for Red Hat OpenShift 1.3.2

Issued: 2023-03-21

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.4.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:1376 - mirror registry for Red Hat OpenShift 1.3.2](#)

4.2.10.11. Mirror registry for Red Hat OpenShift 1.3.1

Issued: 2023-03-7

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.3.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:1086 - mirror registry for Red Hat OpenShift 1.3.1](#)

4.2.10.12. Mirror registry for Red Hat OpenShift 1.3.0

Issued: 2023-02-20

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.8.1.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2023:0558 - mirror registry for Red Hat OpenShift 1.3.0](#)

4.2.10.12.1. New features

- *Mirror registry for Red Hat OpenShift* is now supported on Red Hat Enterprise Linux (RHEL) 9 installations.

- IPv6 support is now available on *mirror registry for Red Hat OpenShift* local host installations. IPv6 is currently unsupported on *mirror registry for Red Hat OpenShift* remote host installations.
- A new feature flag, **--quayStorage**, has been added. By specifying this flag, you can manually set the location for the Quay persistent storage.
- A new feature flag, **--pgStorage**, has been added. By specifying this flag, you can manually set the location for the Postgres persistent storage.
- Previously, users were required to have root privileges (**sudo**) to install *mirror registry for Red Hat OpenShift*. With this update, **sudo** is no longer required to install *mirror registry for Red Hat OpenShift*.

When *mirror registry for Red Hat OpenShift* was installed with **sudo**, an **/etc/quay-install** directory that contained installation files, local storage, and the configuration bundle was created. With the removal of the **sudo** requirement, installation files and the configuration bundle are now installed to **\$HOME/quay-install**. Local storage, for example Postgres and Quay, are now stored in named volumes automatically created by Podman.

To override the default directories that these files are stored in, you can use the command line arguments for *mirror registry for Red Hat OpenShift*. For more information about *mirror registry for Red Hat OpenShift* command line arguments, see "*Mirror registry for Red Hat OpenShift flags*".

4.2.10.12.2. Bug fixes

- Previously, the following error could be returned when attempting to uninstall *mirror registry for Red Hat OpenShift*: **["Error: no container with name or ID \"quay-postgres\" found: no such container"], "stdout": "", "stdout_lines": []***. With this update, the order that *mirror registry for Red Hat OpenShift* services are stopped and uninstalled have been changed so that the error no longer occurs when uninstalling *mirror registry for Red Hat OpenShift*. For more information, see [PROJQUAY-4629](#).

4.2.10.13. Mirror registry for Red Hat OpenShift 1.2.9

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.10.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:7369 - mirror registry for Red Hat OpenShift 1.2.9](#)

4.2.10.14. Mirror registry for Red Hat OpenShift 1.2.8

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.9.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:7065 - mirror registry for Red Hat OpenShift 1.2.8](#)

4.2.10.15. Mirror registry for Red Hat OpenShift 1.2.7

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.8.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:6500 - mirror registry for Red Hat OpenShift 1.2.7](#)

4.2.10.15.1. Bug fixes

- Previously, **getFQDN()** relied on the fully-qualified domain name (FQDN) library to determine its FQDN, and the FQDN library tried to read the **/etc/hosts** folder directly. Consequently, on some Red Hat Enterprise Linux CoreOS (RHCOS) installations with uncommon DNS configurations, the FQDN library would fail to install and abort the installation. With this update, *mirror registry for Red Hat OpenShift* uses **hostname** to determine the FQDN. As a result, the FQDN library does not fail to install. ([PROJQUAY-4139](#))

4.2.10.16. Mirror registry for Red Hat OpenShift 1.2.6

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.7.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:6278 - mirror registry for Red Hat OpenShift 1.2.6](#)

4.2.10.16.1. New features

A new feature flag, **--no-color (-c)** has been added. This feature flag allows users to disable color sequences and propagate that to Ansible when running install, uninstall, and upgrade commands.

4.2.10.17. Mirror registry for Red Hat OpenShift 1.2.5

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.6.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:6071 - mirror registry for Red Hat OpenShift 1.2.5](#)

4.2.10.18. Mirror registry for Red Hat OpenShift 1.2.4

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.5.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:5884 - mirror registry for Red Hat OpenShift 1.2.4](#)

4.2.10.19. Mirror registry for Red Hat OpenShift 1.2.3

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.4.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:5649 - mirror registry for Red Hat OpenShift 1.2.3](#)

4.2.10.20. Mirror registry for Red Hat OpenShift 1.2.2

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.3.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:5501 - mirror registry for Red Hat OpenShift 1.2.2](#)

4.2.10.21. Mirror registry for Red Hat OpenShift 1.2.1

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.2.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:4986 - mirror registry for Red Hat OpenShift 1.2.1](#)

4.2.10.22. Mirror registry for Red Hat OpenShift 1.2.0

Mirror registry for Red Hat OpenShift is now available with Red Hat Quay 3.7.1.

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:4986 - mirror registry for Red Hat OpenShift 1.2.0](#)

4.2.10.22.1. Bug fixes

- Previously, all components and workers running inside of the Quay pod Operator had log levels set to **DEBUG**. As a result, large traffic logs were created that consumed unnecessary space. With this update, log levels are set to **WARN** by default, which reduces traffic information while emphasizing problem scenarios. ([PROJQUAY-3504](#))

4.2.10.23. Mirror registry for Red Hat OpenShift 1.1.0

The following advisory is available for the *mirror registry for Red Hat OpenShift*:

- [RHBA-2022:0956 - mirror registry for Red Hat OpenShift 1.1.0](#)

4.2.10.23.1. New features

- A new command, **mirror-registry upgrade** has been added. This command upgrades all container images without interfering with configurations or data.



NOTE

If **quayRoot** was previously set to something other than default, it must be passed into the upgrade command.

4.2.10.23.2. Bug fixes

- Previously, the absence of **quayHostname** or **targetHostname** did not default to the local hostname. With this update, **quayHostname** and **targetHostname** now default to the local hostname if they are missing. ([PROJQUAY-3079](#))
- Previously, the command `./mirror-registry --version` returned an **unknown flag** error. Now, running `./mirror-registry --version` returns the current version of the *mirror registry for Red Hat OpenShift*. ([PROJQUAY-3086](#))
- Previously, users could not set a password during installation, for example, when running `./mirror-registry install --initUser <user_name> --initPassword <password> --verbose`. With this update, users can set a password during installation. ([PROJQUAY-3149](#))
- Previously, the *mirror registry for Red Hat OpenShift* did not recreate pods if they were destroyed. Now, pods are recreated if they are destroyed. ([PROJQUAY-3261](#))

4.2.11. Troubleshooting mirror registry for Red Hat OpenShift

To assist in troubleshooting *mirror registry for Red Hat OpenShift*, you can gather logs of systemd services installed by the mirror registry. The following services are installed:

- quay-app.service
- quay-postgres.service
- quay-redis.service
- quay-pod.service

Prerequisites

- You have installed *mirror registry for Red Hat OpenShift*.

Procedure

- If you installed *mirror registry for Red Hat OpenShift* with root privileges, you can get the status information of its systemd services by entering the following command:

```
$ sudo systemctl status <service>
```

- If you installed *mirror registry for Red Hat OpenShift* as a standard user, you can get the status information of its systemd services by entering the following command:

```
$ systemctl --user status <service>
```

4.2.12. Additional resources

- [Red Hat Quay garbage collection](#)
- [Using SSL to protect connections to Red Hat Quay](#)
- [Configuring the system to trust the certificate authority](#)
- [Mirroring the OpenShift Container Platform image repository](#)
- [Mirroring Operator catalogs for use with disconnected clusters](#)

4.3. MIRRORING IMAGES FOR A DISCONNECTED INSTALLATION

You can ensure your clusters only use container images that satisfy your organizational controls on external content. Before you install a cluster on infrastructure that you provision in a restricted network, you must mirror the required container images into that environment. To mirror container images, you must have a registry for mirroring.



IMPORTANT

You must have access to the internet to obtain the necessary container images. In this procedure, you place your mirror registry on a mirror host that has access to both your network and the internet. If you do not have access to a mirror host, use the [Mirroring Operator catalogs for use with disconnected clusters](#) procedure to copy images to a device you can move across network boundaries with.

4.3.1. Prerequisites

- You must have a container image registry that supports [Docker v2-2](#) in the location that will host the OpenShift Container Platform cluster, such as one of the following registries:
 - [Red Hat Quay](#)
 - [JFrog Artifactory](#)
 - [Sonatype Nexus Repository](#)
 - [Harbor](#)

If you have an entitlement to Red Hat Quay, see the documentation on deploying Red Hat Quay [for proof-of-concept purposes](#) or [by using the Red Hat Quay Operator](#) . If you need additional assistance selecting and installing a registry, contact your sales representative or Red Hat support.

- If you do not already have an existing solution for a container image registry, subscribers of OpenShift Container Platform are provided a [mirror registry for Red Hat OpenShift](#) . The *mirror registry for Red Hat OpenShift* is included with your subscription and is a small-scale container registry that can be used to mirror the required container images of OpenShift Container Platform in disconnected installations.

4.3.2. About the mirror registry

You can mirror the images that are required for OpenShift Container Platform installation and subsequent product updates to a container mirror registry such as Red Hat Quay, JFrog Artifactory, Sonatype Nexus Repository, or Harbor. If you do not have access to a large-scale container registry, you can use the *mirror registry for Red Hat OpenShift* , a small-scale container registry included with OpenShift Container Platform subscriptions.

You can use any container registry that supports [Docker v2-2](#), such as Red Hat Quay, the *mirror registry for Red Hat OpenShift*, Artifactory, Sonatype Nexus Repository, or Harbor. Regardless of your chosen registry, the procedure to mirror content from Red Hat hosted sites on the internet to an isolated image registry is the same. After you mirror the content, you configure each cluster to retrieve this content from your mirror registry.



IMPORTANT

The OpenShift image registry cannot be used as the target registry because it does not support pushing without a tag, which is required during the mirroring process.

If choosing a container registry that is not the *mirror registry for Red Hat OpenShift*, it must be reachable by every machine in the clusters that you provision. If the registry is unreachable, installation, updating, or normal operations such as workload relocation might fail. For that reason, you must run mirror registries in a highly available way, and the mirror registries must at least match the production availability of your OpenShift Container Platform clusters.

When you populate your mirror registry with OpenShift Container Platform images, you can follow two scenarios. If you have a host that can access both the internet and your mirror registry, but not your cluster nodes, you can directly mirror the content from that machine. This process is referred to as *connected mirroring*. If you have no such host, you must mirror the images to a file system and then bring that host or removable media into your restricted environment. This process is referred to as *disconnected mirroring*.

For mirrored registries, to view the source of pulled images, you must review the **Trying to access** log entry in the CRI-O logs. Other methods to view the image pull source, such as using the **crictl images** command on a node, show the non-mirrored image name, even though the image is pulled from the mirrored location.



NOTE

Red Hat does not test third party registries with OpenShift Container Platform.

Additional information

For information about viewing the CRI-O logs to view the image source, see [Viewing the image pull source](#).

4.3.3. Preparing your mirror host

Before you perform the mirror procedure, you must prepare the host to retrieve content and push it to the remote location.

4.3.3.1. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

4.3.4. Configuring credentials that allow images to be mirrored

Create a container image registry credentials file that allows mirroring images from Red Hat to your mirror.



WARNING

Do not use this image registry credentials file as the pull secret when you install a cluster. If you provide this file when you install cluster, all of the machines in the cluster will have write access to your mirror registry.



WARNING

This process requires that you have write access to a container image registry on the mirror registry and adds the credentials to a registry pull secret.

Prerequisites

- You configured a mirror registry to use in your disconnected environment.
- You identified an image repository location on your mirror registry to mirror images into.
- You provisioned a mirror registry account that allows images to be uploaded to that image repository.

Procedure

Complete the following steps on the installation host:

1. Download your **registry.redhat.io** [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Make a copy of your pull secret in JSON format:

```
$ cat ./pull-secret | jq . > <path>/<pull_secret_file_in_json> 1
```

- 1** Specify the path to the folder to store the pull secret in and a name for the JSON file that you create.

The contents of the file resemble the following example:

```
{
```

```

"auths": {
  "cloud.openshift.com": {
    "auth": "b3BlbnNo...",
    "email": "you@example.com"
  },
  "quay.io": {
    "auth": "b3BlbnNo...",
    "email": "you@example.com"
  },
  "registry.connect.redhat.com": {
    "auth": "NTE3Njg5Nj...",
    "email": "you@example.com"
  },
  "registry.redhat.io": {
    "auth": "NTE3Njg5Nj...",
    "email": "you@example.com"
  }
}
}
}

```

3. Generate the base64-encoded user name and password or token for your mirror registry:

```

$ echo -n '<user_name>:<password>' | base64 -w0 ❶
BGVtbYk3ZHAatqXs=

```

- ❶ For **<user_name>** and **<password>**, specify the user name and password that you configured for your registry.

4. Edit the JSON file and add a section that describes your registry to it:

```

"auths": {
  "<mirror_registry>": { ❶
    "auth": "<credentials>", ❷
    "email": "you@example.com"
  }
},

```

- ❶ For **<mirror_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:8443**
- ❷ For **<credentials>**, specify the base64-encoded user name and password for the mirror registry.

The file resembles the following example:

```

{
  "auths": {
    "registry.example.com": {
      "auth": "BGVtbYk3ZHAatqXs=",
      "email": "you@example.com"
    },
    "cloud.openshift.com": {

```

```

    "auth": "b3BlbnNo...",
    "email": "you@example.com"
  },
  "quay.io": {
    "auth": "b3BlbnNo...",
    "email": "you@example.com"
  },
  "registry.connect.redhat.com": {
    "auth": "NTE3Njg5Nj...",
    "email": "you@example.com"
  },
  "registry.redhat.io": {
    "auth": "NTE3Njg5Nj...",
    "email": "you@example.com"
  }
}
}
}

```

4.3.5. Mirroring the OpenShift Container Platform image repository

Mirror the OpenShift Container Platform image repository to your registry to use during cluster installation or upgrade.

Prerequisites

- Your mirror host has access to the internet.
- You configured a mirror registry to use in your restricted network and can access the certificate and credentials that you configured.
- You downloaded the [pull secret from the Red Hat OpenShift Cluster Manager](#) and modified it to include authentication to your mirror repository.
- If you use self-signed certificates, you have specified a Subject Alternative Name in the certificates.

Procedure

Complete the following steps on the mirror host:

1. Review the [OpenShift Container Platform downloads page](#) to determine the version of OpenShift Container Platform that you want to install and determine the corresponding tag on the [Repository Tags](#) page.
2. Set the required environment variables:
 - a. Export the release version:

```
$ OCP_RELEASE=<release_version>
```

For **<release_version>**, specify the tag that corresponds to the version of OpenShift Container Platform to install, such as **4.5.4**.

- b. Export the local registry name and host port:

```
$ LOCAL_REGISTRY='<local_registry_host_name>:<local_registry_host_port>'
```

For **<local_registry_host_name>**, specify the registry domain name for your mirror repository, and for **<local_registry_host_port>**, specify the port that it serves content on.

- c. Export the local repository name:

```
$ LOCAL_REPOSITORY='<local_repository_name>'
```

For **<local_repository_name>**, specify the name of the repository to create in your registry, such as **ocp4/openshift4**.

- d. Export the name of the repository to mirror:

```
$ PRODUCT_REPO='openshift-release-dev'
```

For a production release, you must specify **openshift-release-dev**.

- e. Export the path to your registry pull secret:

```
$ LOCAL_SECRET_JSON='<path_to_pull_secret>'
```

For **<path_to_pull_secret>**, specify the absolute path to and file name of the pull secret for your mirror registry that you created.

- f. Export the release mirror:

```
$ RELEASE_NAME="ocp-release"
```

For a production release, you must specify **ocp-release**.

- g. Export the type of architecture for your server, such as **x86_64** or **aarch64**:

```
$ ARCHITECTURE=<server_architecture>
```

- h. Export the path to the directory to host the mirrored images:

```
$ REMOVABLE_MEDIA_PATH=<path> 1
```

- 1** Specify the full path, including the initial forward slash (/) character.

3. Mirror the version images to the mirror registry:

- If your mirror host does not have internet access, take the following actions:
 - i. Connect the removable media to a system that is connected to the internet.
 - ii. Review the images and configuration manifests to mirror:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
  --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
  ${ARCHITECTURE} \
  --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
  --to-release-
  image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
  ${ARCHITECTURE} --dry-run
```

- iii. Record the entire **imageContentSources** section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the **imageContentSources** section to the **install-config.yaml** file during installation.
- iv. Mirror the images to a directory on the removable media:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-dir=${REMOVABLE_MEDIA_PATH}/mirror quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-${ARCHITECTURE}
```

- v. Take the media to the restricted network environment and upload the images to the local container registry.

```
$ oc image mirror -a ${LOCAL_SECRET_JSON} --from-dir=${REMOVABLE_MEDIA_PATH}/mirror "file://openshift/release:${OCP_RELEASE}*" ${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} 1
```

- 1** For **REMOVABLE_MEDIA_PATH**, you must use the same path that you specified when you mirrored the images.



IMPORTANT

Running **oc image mirror** might result in the following error: **error: unable to retrieve source image**. This error occurs when image indexes include references to images that no longer exist on the image registry. Image indexes might retain older references to allow users running those images an upgrade path to newer points on the upgrade graph. As a temporary workaround, you can use the **--skip-missing** option to bypass the error and continue downloading the image index. For more information, see [Service Mesh Operator mirroring failed](#).

- If the local container registry is connected to the mirror host, take the following actions:
 - i. Directly push the release images to the local registry by using following command:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
  --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-${ARCHITECTURE} \
  --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
  --to-release-image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-${ARCHITECTURE}
```

This command pulls the release information as a digest, and its output includes the **imageContentSources** data that you require when you install your cluster.

- ii. Record the entire **imageContentSources** section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the **imageContentSources** section to the **install-config.yaml** file during installation.

**NOTE**

The image name gets patched to Quay.io during the mirroring process, and the podman images will show Quay.io in the registry on the bootstrap virtual machine.

- To create the installation program that is based on the content that you mirrored, extract it and pin it to the release:

- If your mirror host does not have internet access, run the following command:

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --icsp-file=<file> \ --
command=openshift-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}"
```

- If the local container registry is connected to the mirror host, run the following command:

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}"
```

**IMPORTANT**

To ensure that you use the correct images for the version of OpenShift Container Platform that you selected, you must extract the installation program from the mirrored content.

You must perform this step on a machine with an active internet connection.

- For clusters using installer-provisioned infrastructure, run the following command:

```
$ openshift-install
```

4.3.6. The Cluster Samples Operator in a disconnected environment

In a disconnected environment, you must take additional steps after you install a cluster to configure the Cluster Samples Operator. Review the following information in preparation.

4.3.6.1. Cluster Samples Operator assistance for mirroring

During installation, OpenShift Container Platform creates a config map named **imagestreamtag-to-image** in the **openshift-cluster-samples-operator** namespace. The **imagestreamtag-to-image** config map contains an entry, the populating image, for each image stream tag.

The format of the key for each entry in the data field in the config map is **<image_stream_name>_<image_stream_tag_name>**.

During a disconnected installation of OpenShift Container Platform, the status of the Cluster Samples Operator is set to **Removed**. If you choose to change it to **Managed**, it installs samples.

**NOTE**

The use of samples in a network-restricted or discontinued environment may require access to services external to your network. Some example services include: Github, Maven Central, npm, RubyGems, PyPi and others. There might be additional steps to take that allow the cluster samples operators's objects to reach the services they require.

You can use this config map as a reference for which images need to be mirrored for your image streams to import.

- While the Cluster Samples Operator is set to **Removed**, you can create your mirrored registry, or determine which existing mirrored registry you want to use.
- Mirror the samples you want to the mirrored registry using the new config map as your guide.
- Add any of the image streams you did not mirror to the **skippedImagestreams** list of the Cluster Samples Operator configuration object.
- Set **samplesRegistry** of the Cluster Samples Operator configuration object to the mirrored registry.
- Then set the Cluster Samples Operator to **Managed** to install the image streams you have mirrored.

4.3.7. Mirroring Operator catalogs for use with disconnected clusters

You can mirror the Operator contents of a Red Hat-provided catalog, or a custom catalog, into a container image registry using the **oc adm catalog mirror** command. The target registry must support [Docker v2-2](#). For a cluster on a restricted network, this registry can be one that the cluster has network access to, such as a mirror registry created during a restricted network cluster installation.

**IMPORTANT**

- The OpenShift image registry cannot be used as the target registry because it does not support pushing without a tag, which is required during the mirroring process.
- Running **oc adm catalog mirror** might result in the following error: **error: unable to retrieve source image**. This error occurs when image indexes include references to images that no longer exist on the image registry. Image indexes might retain older references to allow users running those images an upgrade path to newer points on the upgrade graph. As a temporary workaround, you can use the **--skip-missing** option to bypass the error and continue downloading the image index. For more information, see [Service Mesh Operator mirroring failed](#).

The **oc adm catalog mirror** command also automatically mirrors the index image that is specified during the mirroring process, whether it be a Red Hat-provided index image or your own custom-built index image, to the target registry. You can then use the mirrored index image to create a catalog source that allows Operator Lifecycle Manager (OLM) to load the mirrored catalog onto your OpenShift Container Platform cluster.

Additional resources

- [Using Operator Lifecycle Manager on restricted networks](#)

4.3.7.1. Prerequisites

Mirroring Operator catalogs for use with disconnected clusters has the following prerequisites:

- Workstation with unrestricted network access.
- **podman** version 1.9.3 or later.
- If you want to filter, or *prune*, an existing catalog and selectively mirror only a subset of Operators, see the following sections:
 - [Installing the opm CLI](#)
 - [Updating or filtering a file-based catalog image](#)

- If you want to mirror a Red Hat–provided catalog, run the following command on your workstation with unrestricted network access to authenticate with **registry.redhat.io**:

```
$ podman login registry.redhat.io
```

- Access to a mirror registry that supports [Docker v2-2](#).
- On your mirror registry, decide which repository, or namespace, to use for storing mirrored Operator content. For example, you might create an **olm-mirror** repository.
- If your mirror registry does not have internet access, connect removable media to your workstation with unrestricted network access.
- If you are working with private registries, including **registry.redhat.io**, set the **REG_CREDS** environment variable to the file path of your registry credentials for use in later steps. For example, for the **podman** CLI:

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

4.3.7.2. Extracting and mirroring catalog contents

The **oc adm catalog mirror** command extracts the contents of an index image to generate the manifests required for mirroring. The default behavior of the command generates manifests, then automatically mirrors all of the image content from the index image, as well as the index image itself, to your mirror registry.

Alternatively, if your mirror registry is on a completely disconnected, or *airgapped*, host, you can first mirror the content to removable media, move the media to the disconnected environment, then mirror the content from the media to the registry.

4.3.7.2.1. Mirroring catalog contents to registries on the same network

If your mirror registry is co-located on the same network as your workstation with unrestricted network access, take the following actions on your workstation.

Procedure

1. If your mirror registry requires authentication, run the following command to log in to the registry:

```
$ podman login <mirror_registry>
```

2. Run the following command to extract and mirror the content to the mirror registry:

```
$ oc adm catalog mirror \
  <index_image> \ 1
  <mirror_registry>:<port>[/<repository>] \ 2
  [-a ${REG_CREDS}] \ 3
  [--insecure] \ 4
  [--index-filter-by-os='<platform>/<arch>'] \ 5
  [--manifests-only] 6
```

- 1 Specify the index image for the catalog that you want to mirror.
- 2 Specify the fully qualified domain name (FQDN) for the target registry to mirror the Operator contents to. The mirror registry **<repository>** can be any existing repository, or namespace, on the registry, for example **olm-mirror** as outlined in the prerequisites. If there is an existing repository found during mirroring, the repository name is added to the resulting image name. If you do not want the image name to include the repository name, omit the **<repository>** value from this line, for example **<mirror_registry>:<port>**.
- 3 Optional: If required, specify the location of your registry credentials file. **{REG_CREDS}** is required for **registry.redhat.io**.
- 4 Optional: If you do not want to configure trust for the target registry, add the **--insecure** flag.
- 5 Optional: Specify which platform and architecture of the index image to select when multiple variants are available. Images are passed as '**<platform>/<arch>[/<variant>]**'. This does not apply to images referenced by the index. Valid values are **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**.
- 6 Optional: Generate only the manifests required for mirroring without actually mirroring the image content to a registry. This option can be useful for reviewing what will be mirrored, and lets you make any changes to the mapping list, if you require only a subset of packages. You can then use the **mapping.txt** file with the **oc image mirror** command to mirror the modified list of images in a later step. This flag is intended for only advanced selective mirroring of content from the catalog.

Example output

```
src image has index label for database path: /database/index.db
using database path mapping: /database/index.db:/tmp/153048078
wrote database to /tmp/153048078 1
...
wrote mirroring manifests to manifests-redhat-operator-index-1614211642 2
```

- 1 Directory for the temporary **index.db** database generated by the command.
- 2 Record the manifests directory name that is generated. This directory is referenced in subsequent procedures.



NOTE

Red Hat Quay does not support nested repositories. As a result, running the **oc adm catalog mirror** command will fail with a **401** unauthorized error. As a workaround, you can use the **--max-components=2** option when running the **oc adm catalog mirror** command to disable the creation of nested repositories. For more information on this workaround, see the [Unauthorized error thrown while using catalog mirror command with Quay registry](#) Knowledgebase Solution.

Additional resources

- [Architecture and operating system support for Operators](#)

4.3.7.2.2. Mirroring catalog contents to airgapped registries

If your mirror registry is on a completely disconnected, or airgapped, host, take the following actions.

Procedure

1. Run the following command on your workstation with unrestricted network access to mirror the content to local files:

```
$ oc adm catalog mirror \
  <index_image> \ 1
  file:///local/index \ 2
  -a ${REG_CREDS} \ 3
  --insecure \ 4
  --index-filter-by-os='<platform>/<arch>' 5
```

- 1** Specify the index image for the catalog that you want to mirror.
- 2** Specify the content to mirror to local files in your current directory.
- 3** Optional: If required, specify the location of your registry credentials file.
- 4** Optional: If you do not want to configure trust for the target registry, add the **--insecure** flag.
- 5** Optional: Specify which platform and architecture of the index image to select when multiple variants are available. Images are specified as '**<platform>/<arch>/<variant>**'. This does not apply to images referenced by the index. Valid values are **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**, and *****.

Example output

```
...
info: Mirroring completed in 5.93s (5.915MB/s)
wrote mirroring manifests to manifests-my-index-1614985528 1
```

To upload local images to a registry, run:

```
oc adm catalog mirror file:///local/index/myrepo/my-index:v1 REGISTRY/REPOSITORY 2
```

- 1** Record the manifests directory name that is generated. This directory is referenced in subsequent procedures.
- 2** Record the expanded **file://** path that is based on your provided index image. This path is referenced in a subsequent step.

This command creates a **v2/** directory in your current directory.

2. Copy the **v2/** directory to removable media.
3. Physically remove the media and attach it to a host in the disconnected environment that has access to the mirror registry.
4. If your mirror registry requires authentication, run the following command on your host in the disconnected environment to log in to the registry:

```
$ podman login <mirror_registry>
```

5. Run the following command from the parent directory containing the **v2/** directory to upload the images from local files to the mirror registry:

```
$ oc adm catalog mirror \  
  file:///local/index/<repository>/<index_image>:<tag> \1 \  
  <mirror_registry>:<port>[/<repository>] \2 \  
  -a ${REG_CREDS} \3 \  
  --insecure \4 \  
  --index-filter-by-os='<platform>/<arch>' 5
```

- 1** Specify the **file://** path from the previous command output.
- 2** Specify the fully qualified domain name (FQDN) for the target registry to mirror the Operator contents to. The mirror registry **<repository>** can be any existing repository, or namespace, on the registry, for example **olm-mirror** as outlined in the prerequisites. If there is an existing repository found during mirroring, the repository name is added to the resulting image name. If you do not want the image name to include the repository name, omit the **<repository>** value from this line, for example **<mirror_registry>:<port>**.
- 3** Optional: If required, specify the location of your registry credentials file.
- 4** Optional: If you do not want to configure trust for the target registry, add the **--insecure** flag.
- 5** Optional: Specify which platform and architecture of the index image to select when multiple variants are available. Images are specified as **'<platform>/<arch>[/<variant>]'**. This does not apply to images referenced by the index. Valid values are **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**, and *****.



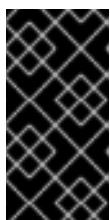
NOTE

Red Hat Quay does not support nested repositories. As a result, running the **oc adm catalog mirror** command will fail with a **401** unauthorized error. As a workaround, you can use the **--max-components=2** option when running the **oc adm catalog mirror** command to disable the creation of nested repositories. For more information on this workaround, see the [Unauthorized error thrown while using catalog mirror command with Quay registry](#) Knowledgebase Solution.

- Run the **oc adm catalog mirror** command again. Use the newly mirrored index image as the source and the same mirror registry target used in the previous step:

```
$ oc adm catalog mirror \
  <mirror_registry>:<port>/<index_image> \
  <mirror_registry>:<port>/[<repository>] \
  --manifests-only 1 \
  [-a ${REG_CREDS}] \
  [--insecure]
```

- The **--manifests-only** flag is required for this step so that the command does not copy all of the mirrored content again.



IMPORTANT

This step is required because the image mappings in the **imageContentSourcePolicy.yaml** file generated during the previous step must be updated from local paths to valid mirror locations. Failure to do so will cause errors when you create the **ImageContentSourcePolicy** object in a later step.

After you mirror the catalog, you can continue with the remainder of your cluster installation. After your cluster installation has finished successfully, you must specify the manifests directory from this procedure to create the **ImageContentSourcePolicy** and **CatalogSource** objects. These objects are required to enable installation of Operators from OperatorHub.

Additional resources

- [Architecture and operating system support for Operators](#)

4.3.7.3. Generated manifests

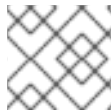
After mirroring Operator catalog content to your mirror registry, a manifests directory is generated in your current directory.

If you mirrored content to a registry on the same network, the directory name takes the following pattern:

```
manifests-<index_image_name>-<random_number>
```

If you mirrored content to a registry on a disconnected host in the previous section, the directory name takes the following pattern:

```
manifests-index/<repository>/<index_image_name>-<random_number>
```

**NOTE**

The manifests directory name is referenced in subsequent procedures.

The manifests directory contains the following files, some of which might require further modification:

- The **catalogSource.yaml** file is a basic definition for a **CatalogSource** object that is pre-populated with your index image tag and other relevant metadata. This file can be used as is or modified to add the catalog source to your cluster.

**IMPORTANT**

If you mirrored the content to local files, you must modify your **catalogSource.yaml** file to remove any backslash (/) characters from the **metadata.name** field. Otherwise, when you attempt to create the object, it fails with an "invalid resource name" error.

- The **imageContentSourcePolicy.yaml** file defines an **ImageContentSourcePolicy** object that can configure nodes to translate between the image references stored in Operator manifests and the mirrored registry.

**NOTE**

If your cluster uses an **ImageContentSourcePolicy** object to configure repository mirroring, you can use only global pull secrets for mirrored registries. You cannot add a pull secret to a project.

- The **mapping.txt** file contains all of the source images and where to map them in the target registry. This file is compatible with the **oc image mirror** command and can be used to further customize the mirroring configuration.

**IMPORTANT**

If you used the **--manifests-only** flag during the mirroring process and want to further trim the subset of packages to mirror, see the steps in the [Mirroring a package manifest format catalog image](#) procedure of the OpenShift Container Platform 4.7 documentation about modifying your **mapping.txt** file and using the file with the **oc image mirror** command.

4.3.7.4. Postinstallation requirements

After you mirror the catalog, you can continue with the remainder of your cluster installation. After your cluster installation has finished successfully, you must specify the manifests directory from this procedure to create the **ImageContentSourcePolicy** and **CatalogSource** objects. These objects are required to populate and enable installation of Operators from OperatorHub.

Additional resources

- [Populating OperatorHub from mirrored Operator catalogs](#)
- [Updating or filtering a file-based catalog image](#)

4.3.8. Next steps

- Install a cluster on infrastructure that you provision in your restricted network, such as on [VMware vSphere](#), [bare metal](#), or [Amazon Web Services](#).

4.3.9. Additional resources

- See [Gathering data about specific features](#) for more information about using must-gather.

4.4. MIRRORING IMAGES FOR A DISCONNECTED INSTALLATION USING THE OC-MIRROR PLUGIN

Running your cluster in a restricted network without direct internet connectivity is possible by installing the cluster from a mirrored set of OpenShift Container Platform container images in a private registry. This registry must be running at all times as long as the cluster is running. See the [Prerequisites](#) section for more information.

You can use the oc-mirror OpenShift CLI (**oc**) plugin to mirror images to a mirror registry in your fully or partially disconnected environments. You must run oc-mirror from a system with internet connectivity in order to download the required images from the official Red Hat registries.

The following steps outline the high-level workflow on how to use the oc-mirror plugin to mirror images to a mirror registry:

1. Create an image set configuration file.
2. Mirror the image set to the mirror registry by using one of the following methods:
 - Mirror an image set directly to the mirror registry.
 - Mirror an image set to disk, transfer the image set to the target environment, then upload the image set to the target mirror registry.
3. Configure your cluster to use the resources generated by the oc-mirror plugin.
4. Repeat these steps to update your mirror registry as necessary.

4.4.1. About the oc-mirror plugin

You can use the oc-mirror OpenShift CLI (**oc**) plugin to mirror all required OpenShift Container Platform content and other images to your mirror registry by using a single tool. It provides the following features:

- Provides a centralized method to mirror OpenShift Container Platform releases, Operators, helm charts, and other images.
- Maintains update paths for OpenShift Container Platform and Operators.
- Uses a declarative image set configuration file to include only the OpenShift Container Platform releases, Operators, and images that your cluster needs.
- Performs incremental mirroring, which reduces the size of future image sets.
- Prunes images from the target mirror registry that were excluded from the image set configuration since the previous execution.
- Optionally generates supporting artifacts for OpenShift Update Service (OSUS) usage.

When using the `oc-mirror` plugin, you specify which content to mirror in an image set configuration file. In this YAML file, you can fine-tune the configuration to only include the OpenShift Container Platform releases and Operators that your cluster needs. This reduces the amount of data that you need to download and transfer. The `oc-mirror` plugin can also mirror arbitrary helm charts and additional container images to assist users in seamlessly synchronizing their workloads onto mirror registries.

The first time you run the `oc-mirror` plugin, it populates your mirror registry with the required content to perform your disconnected cluster installation or update. In order for your disconnected cluster to continue receiving updates, you must keep your mirror registry updated. To update your mirror registry, you run the `oc-mirror` plugin using the same configuration as the first time you ran it. The `oc-mirror` plugin references the metadata from the storage backend and only downloads what has been released since the last time you ran the tool. This provides update paths for OpenShift Container Platform and Operators and performs dependency resolution as required.



IMPORTANT

When using the `oc-mirror` CLI plugin to populate a mirror registry, any further updates to the mirror registry must be made using the `oc-mirror` tool.

4.4.2. `oc-mirror` compatibility and support

The `oc-mirror` plugin supports mirroring OpenShift Container Platform payload images and Operator catalogs for OpenShift Container Platform versions 4.9 and later.

Use the latest available version of the `oc-mirror` plugin regardless of which versions of OpenShift Container Platform you need to mirror.

4.4.3. About the mirror registry

You can mirror the images that are required for OpenShift Container Platform installation and subsequent product updates to a container mirror registry that supports [Docker v2-2](#), such as Red Hat Quay. If you do not have access to a large-scale container registry, you can use the *mirror registry for Red Hat OpenShift*, which is a small-scale container registry included with OpenShift Container Platform subscriptions.

Regardless of your chosen registry, the procedure to mirror content from Red Hat hosted sites on the internet to an isolated image registry is the same. After you mirror the content, you configure each cluster to retrieve this content from your mirror registry.



IMPORTANT

The OpenShift image registry cannot be used as the target registry because it does not support pushing without a tag, which is required during the mirroring process.

If choosing a container registry that is not the *mirror registry for Red Hat OpenShift*, it must be reachable by every machine in the clusters that you provision. If the registry is unreachable, installation, updating, or normal operations such as workload relocation might fail. For that reason, you must run mirror registries in a highly available way, and the mirror registries must at least match the production availability of your OpenShift Container Platform clusters.

When you populate your mirror registry with OpenShift Container Platform images, you can follow two scenarios. If you have a host that can access both the internet and your mirror registry, but not your cluster nodes, you can directly mirror the content from that machine. This process is referred to as

connected mirroring. If you have no such host, you must mirror the images to a file system and then bring that host or removable media into your restricted environment. This process is referred to as *disconnected mirroring*.

For mirrored registries, to view the source of pulled images, you must review the **Trying to access** log entry in the CRI-O logs. Other methods to view the image pull source, such as using the **crictl images** command on a node, show the non-mirrored image name, even though the image is pulled from the mirrored location.



NOTE

Red Hat does not test third party registries with OpenShift Container Platform.

Additional resources

- For information about viewing the CRI-O logs to view the image source, see [Viewing the image pull source](#).

4.4.4. Prerequisites

- You must have a container image registry that supports [Docker v2-2](#) in the location that will host the OpenShift Container Platform cluster, such as Red Hat Quay.



NOTE

If you use Red Hat Quay, you must use version 3.6 or later with the `oc-mirror` plugin. If you have an entitlement to Red Hat Quay, see the documentation on deploying Red Hat Quay [for proof-of-concept purposes](#) or [by using the Red Hat Quay Operator](#). If you need additional assistance selecting and installing a registry, contact your sales representative or Red Hat Support.

If you do not already have an existing solution for a container image registry, subscribers of OpenShift Container Platform are provided a [mirror registry for Red Hat OpenShift](#). The *mirror registry for Red Hat OpenShift* is included with your subscription and is a small-scale container registry that can be used to mirror the required container images of OpenShift Container Platform in disconnected installations.

4.4.5. Preparing your mirror hosts

Before you can use the `oc-mirror` plugin to mirror images, you must install the plugin and create a container image registry credentials file to allow the mirroring from Red Hat to your mirror.

4.4.5.1. Installing the `oc-mirror` OpenShift CLI plugin

To use the `oc-mirror` OpenShift CLI plugin to mirror registry images, you must install the plugin. If you are mirroring image sets in a fully disconnected environment, ensure that you install the `oc-mirror` plugin on the host with internet access and the host in the disconnected environment with access to the mirror registry.

Prerequisites

- You have installed the OpenShift CLI (**oc**).

Procedure

1. Download the oc-mirror CLI plugin.
 - a. Navigate to the [Downloads](#) page of the [OpenShift Cluster Manager Hybrid Cloud Console](#).
 - b. Under the **OpenShift disconnected installation tools** section, click **Download** for **OpenShift Client (oc) mirror plugin** and save the file.
2. Extract the archive:

```
$ tar xvzf oc-mirror.tar.gz
```

3. If necessary, update the plugin file to be executable:

```
$ chmod +x oc-mirror
```



NOTE

Do not rename the **oc-mirror** file.

4. Install the oc-mirror CLI plugin by placing the file in your **PATH**, for example, **/usr/local/bin**:

```
$ sudo mv oc-mirror /usr/local/bin/.
```

Verification

- Run **oc mirror help** to verify that the plugin was successfully installed:

```
$ oc mirror help
```

Additional resources

- [Installing and using CLI plugins](#)

4.4.5.2. Configuring credentials that allow images to be mirrored

Create a container image registry credentials file that allows mirroring images from Red Hat to your mirror.



WARNING

Do not use this image registry credentials file as the pull secret when you install a cluster. If you provide this file when you install cluster, all of the machines in the cluster will have write access to your mirror registry.

**WARNING**

This process requires that you have write access to a container image registry on the mirror registry and adds the credentials to a registry pull secret.

Prerequisites

- You configured a mirror registry to use in your disconnected environment.
- You identified an image repository location on your mirror registry to mirror images into.
- You provisioned a mirror registry account that allows images to be uploaded to that image repository.

Procedure

Complete the following steps on the installation host:

1. Download your **registry.redhat.io** pull secret from the [Red Hat OpenShift Cluster Manager](#) .
2. Make a copy of your pull secret in JSON format:

```
$ cat ./pull-secret | jq . > <path>/<pull_secret_file_in_json> 1
```

- 1 Specify the path to the folder to store the pull secret in and a name for the JSON file that you create.

The contents of the file resemble the following example:

```
{
  "auths": {
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
```

3. Save the file as either `~/.docker/config.json` or `$XDG_RUNTIME_DIR/containers/auth.json`:

- a. If the **.docker** or **\$XDG_RUNTIME_DIR/containers** directories do not exist, create one by entering the following command:

```
$ mkdir -p <directory_name>
```

Where **<directory_name>** is either **~/.docker** or **\$XDG_RUNTIME_DIR/containers**.

- b. Copy the pull secret to the appropriate directory by entering the following command:

```
$ cp <path>/<pull_secret_file_in_json> <directory_name>/<auth_file>
```

Where **<directory_name>** is either **~/.docker** or **\$XDG_RUNTIME_DIR/containers**, and **<auth_file>** is either **config.json** or **auth.json**.

4. Generate the base64-encoded user name and password or token for your mirror registry:

```
$ echo -n '<user_name>:<password>' | base64 -w0 1  
BGVtbYk3ZHAqXs=
```

- 1** For **<user_name>** and **<password>**, specify the user name and password that you configured for your registry.

5. Edit the JSON file and add a section that describes your registry to it:

```
"auths": {  
  "<mirror_registry>": { 1  
    "auth": "<credentials>", 2  
    "email": "you@example.com"  
  }  
},
```

- 1** For **<mirror_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:8443**

- 2** For **<credentials>**, specify the base64-encoded user name and password for the mirror registry.

The file resembles the following example:

```
{  
  "auths": {  
    "registry.example.com": {  
      "auth": "BGVtbYk3ZHAqXs=",  
      "email": "you@example.com"  
    },  
    "cloud.openshift.com": {  
      "auth": "b3BlbnNo...",  
      "email": "you@example.com"  
    },  
    "quay.io": {  
      "auth": "b3BlbnNo...",  
      "email": "you@example.com"  
    }  
  }  
}
```

```

    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
}

```

4.4.6. Creating the image set configuration

Before you can use the `oc-mirror` plugin to mirror image sets, you must create an image set configuration file. This image set configuration file defines which OpenShift Container Platform releases, Operators, and other images to mirror, along with other configuration settings for the `oc-mirror` plugin.

You must specify a storage backend in the image set configuration file. This storage backend can be a local directory or a registry that supports [Docker v2-2](#). The `oc-mirror` plugin stores metadata in this storage backend during image set creation.



IMPORTANT

Do not delete or modify the metadata that is generated by the `oc-mirror` plugin. You must use the same storage backend every time you run the `oc-mirror` plugin for the same mirror registry.

Prerequisites

- You have created a container image registry credentials file. For instructions, see *Configuring credentials that allow images to be mirrored*.

Procedure

- Use the **`oc mirror init`** command to create a template for the image set configuration and save it to a file called **`imageset-config.yaml`**:

```
$ oc mirror init --registry example.com/mirror/oc-mirror-metadata > imageset-config.yaml 1
```

- 1** Replace **`example.com/mirror/oc-mirror-metadata`** with the location of your registry for the storage backend.

- Edit the file and adjust the settings as necessary:

```

kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4 1
storageConfig: 2
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata 3
    skipTLS: false

```

```

mirror:
  platform:
    channels:
      - name: stable-4.12
      type: ocp
    graph: true
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
    packages:
      - name: serverless-operator
        channels:
          - name: stable
  additionalImages:
    - name: registry.redhat.io/ubi8/ubi:latest
  helm: {}

```

- 1 Add **archiveSize** to set the maximum size, in GiB, of each file within the image set.
- 2 Set the back-end location to save the image set metadata to. This location can be a registry or local directory. It is required to specify **storageConfig** values, unless you are using the Technology Preview OCI feature.
- 3 Set the registry URL for the storage backend.
- 4 Set the channel to retrieve the OpenShift Container Platform images from.
- 5 Add **graph: true** to build and push the graph-data image to the mirror registry. The graph-data image is required to create OpenShift Update Service (OSUS). The **graph: true** field also generates the **UpdateService** custom resource manifest. The **oc** command-line interface (CLI) can use the **UpdateService** custom resource manifest to create OSUS. For more information, see *About the OpenShift Update Service*.
- 6 Set the Operator catalog to retrieve the OpenShift Container Platform images from.
- 7 Specify only certain Operator packages to include in the image set. Remove this field to retrieve all packages in the catalog.
- 8 Specify only certain channels of the Operator packages to include in the image set. You must always include the default channel for the Operator package even if you do not use the bundles in that channel. You can find the default channel by running the following command: **oc mirror list operators --catalog=<catalog_name> --package=<package_name>**.
- 9 Specify any additional images to include in image set.



NOTE

The **graph: true** field also mirrors the **ubi-micro** image along with other mirrored images.

See *Image set configuration parameters* for the full list of parameters and *Image set configuration examples* for various mirroring use cases.

3. Save the updated file.

This image set configuration file is required by the **oc mirror** command when mirroring content.

Additional resources

- [Image set configuration parameters](#)
- [Image set configuration examples](#)
- [Using the OpenShift Update Service in a disconnected environment](#)

4.4.7. Mirroring an image set to a mirror registry

You can use the `oc-mirror` CLI plugin to mirror images to a mirror registry in a [partially disconnected environment](#) or in a [fully disconnected environment](#).

These procedures assume that you already have your mirror registry set up.

4.4.7.1. Mirroring an image set in a partially disconnected environment

In a partially disconnected environment, you can mirror an image set directly to the target mirror registry.

4.4.7.1.1. Mirroring from mirror to mirror

You can use the `oc-mirror` plugin to mirror an image set directly to a target mirror registry that is accessible during image set creation.

You are required to specify a storage backend in the image set configuration file. This storage backend can be a local directory or a Docker v2 registry. The `oc-mirror` plugin stores metadata in this storage backend during image set creation.



IMPORTANT

Do not delete or modify the metadata that is generated by the `oc-mirror` plugin. You must use the same storage backend every time you run the `oc-mirror` plugin for the same mirror registry.

Prerequisites

- You have access to the internet to obtain the necessary container images.
- You have installed the OpenShift CLI (**oc**).
- You have installed the **oc-mirror** CLI plugin.
- You have created the image set configuration file.

Procedure

- Run the **oc mirror** command to mirror the images from the specified image set configuration to a specified registry:

```
$ oc mirror --config=./imageset-config.yaml \ 1  
docker://registry.example:5000 2
```


- 1 Pass in the image set configuration file that was created. This procedure assumes that it is named **imageset-config.yaml**.
- 2 Specify the registry to mirror the image set file to. The registry must start with **docker://**. If you specify a top-level namespace for the mirror registry, you must also use this same namespace on subsequent executions.

Verification

1. Navigate into the **oc-mirror-workspace/** directory that was generated.
2. Navigate into the results directory, for example, **results-1639608409/**.
3. Verify that YAML files are present for the **ImageContentSourcePolicy** and **CatalogSource** resources.

Next steps

- Configure your cluster to use the resources generated by oc-mirror.

Troubleshooting

- [Unable to retrieve source image](#) .

4.4.7.2. Mirroring an image set in a fully disconnected environment

To mirror an image set in a fully disconnected environment, you must first [mirror the image set to disk](#) , then [mirror the image set file on disk to a mirror](#) .

4.4.7.2.1. Mirroring from mirror to disk

You can use the oc-mirror plugin to generate an image set and save the contents to disk. The generated image set can then be transferred to the disconnected environment and mirrored to the target registry.

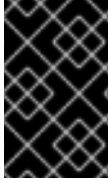


IMPORTANT

Depending on the configuration specified in the image set configuration file, using oc-mirror to mirror images might download several hundreds of gigabytes of data to disk.

The initial image set download when you populate the mirror registry is often the largest. Because you only download the images that changed since the last time you ran the command, when you run the oc-mirror plugin again, the generated image set is often smaller.

You are required to specify a storage backend in the image set configuration file. This storage backend can be a local directory or a docker v2 registry. The oc-mirror plugin stores metadata in this storage backend during image set creation.



IMPORTANT

Do not delete or modify the metadata that is generated by the `oc-mirror` plugin. You must use the same storage backend every time you run the `oc-mirror` plugin for the same mirror registry.

Prerequisites

- You have access to the internet to obtain the necessary container images.
- You have installed the OpenShift CLI (**oc**).
- You have installed the **oc-mirror** CLI plugin.
- You have created the image set configuration file.

Procedure

- Run the **oc mirror** command to mirror the images from the specified image set configuration to disk:

```
$ oc mirror --config=./imageset-config.yaml \ 1  
file://<path_to_output_directory> 2
```

- 1 Pass in the image set configuration file that was created. This procedure assumes that it is named **imageset-config.yaml**.
- 2 Specify the target directory where you want to output the image set file. The target directory path must start with **file://**.

Verification

1. Navigate to your output directory:

```
$ cd <path_to_output_directory>
```

2. Verify that an image set **.tar** file was created:

```
$ ls
```

Example output

```
mirror_seq1_000000.tar
```

Next steps

- Transfer the image set **.tar** file to the disconnected environment.

Troubleshooting

- [Unable to retrieve source image](#) .

4.4.7.2.2. Mirroring from disk to mirror

You can use the `oc-mirror` plugin to mirror the contents of a generated image set to the target mirror registry.

Prerequisites

- You have installed the OpenShift CLI (**oc**) in the disconnected environment.
- You have installed the **oc-mirror** CLI plugin in the disconnected environment.
- You have generated the image set file by using the **oc mirror** command.
- You have transferred the image set file to the disconnected environment.

Procedure

- Run the **oc mirror** command to process the image set file on disk and mirror the contents to a target mirror registry:

```
$ oc mirror --from=./mirror_seq1_000000.tar \ 1
docker://registry.example:5000 2
```

- 1 Pass in the image set `.tar` file to mirror, named **mirror_seq1_000000.tar** in this example. If an **archiveSize** value was specified in the image set configuration file, the image set might be broken up into multiple `.tar` files. In this situation, you can pass in a directory that contains the image set `.tar` files.
- 2 Specify the registry to mirror the image set file to. The registry must start with **docker://**. If you specify a top-level namespace for the mirror registry, you must also use this same namespace on subsequent executions.

This command updates the mirror registry with the image set and generates the **ImageContentSourcePolicy** and **CatalogSource** resources.

Verification

1. Navigate into the **oc-mirror-workspace/** directory that was generated.
2. Navigate into the results directory, for example, **results-1639608409/**.
3. Verify that YAML files are present for the **ImageContentSourcePolicy** and **CatalogSource** resources.

Next steps

- Configure your cluster to use the resources generated by `oc-mirror`.

Troubleshooting

- [Unable to retrieve source image](#) .

4.4.8. Configuring your cluster to use the resources generated by oc-mirror

After you have mirrored your image set to the mirror registry, you must apply the generated **ImageContentSourcePolicy**, **CatalogSource**, and release image signature resources into the cluster.

The **ImageContentSourcePolicy** resource associates the mirror registry with the source registry and redirects image pull requests from the online registries to the mirror registry. The **CatalogSource** resource is used by Operator Lifecycle Manager (OLM) to retrieve information about the available Operators in the mirror registry. The release image signatures are used to verify the mirrored release images.

Prerequisites

- You have mirrored the image set to the registry mirror in the disconnected environment.
- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Log in to the OpenShift CLI as a user with the **cluster-admin** role.
2. Apply the YAML files from the results directory to the cluster by running the following command:

```
$ oc apply -f ./oc-mirror-workspace/results-1639608409/
```

3. If you mirrored release images, apply the release image signatures to the cluster by running the following command:

```
$ oc apply -f ./oc-mirror-workspace/results-1639608409/release-signatures/
```



NOTE

If you are mirroring Operators instead of clusters, you do not need to run **\$ oc apply -f ./oc-mirror-workspace/results-1639608409/release-signatures/**. Running that command will return an error, as there are no release image signatures to apply.

Verification

1. Verify that the **ImageContentSourcePolicy** resources were successfully installed by running the following command:

```
$ oc get imagecontentsourcepolicy
```

2. Verify that the **CatalogSource** resources were successfully installed by running the following command:

```
$ oc get catalogsource -n openshift-marketplace
```

4.4.9. Keeping your mirror registry content updated

After your target mirror registry is populated with the initial image set, be sure to update it regularly so that it has the latest content. You can optionally set up a cron job, if possible, so that the mirror registry is updated on a regular basis.

Ensure that you update your image set configuration to add or remove OpenShift Container Platform and Operator releases as necessary. Any images that are removed are pruned from the mirror registry.

4.4.9.1. About updating your mirror registry content

When you run the `oc-mirror` plugin again, it generates an image set that only contains new and updated images since the previous execution. Because it only pulls in the differences since the previous image set was created, the generated image set is often smaller and faster to process than the initial image set.



IMPORTANT

Generated image sets are sequential and must be pushed to the target mirror registry in order. You can derive the sequence number from the file name of the generated image set archive file.

Adding new and updated images

Depending on the settings in your image set configuration, future executions of `oc-mirror` can mirror additional new and updated images. Review the settings in your image set configuration to ensure that you are retrieving new versions as necessary. For example, you can set the minimum and maximum versions of Operators to mirror if you want to restrict to specific versions. Alternatively, you can set the minimum version as a starting point to mirror, but keep the version range open so you keep receiving new Operator versions on future executions of `oc-mirror`. Omitting any minimum or maximum version gives you the full version history of an Operator in a channel. Omitting explicitly named channels gives you all releases in all channels of the specified Operator. Omitting any named Operator gives you the entire catalog of all Operators and all their versions ever released.

All these constraints and conditions are evaluated against the publicly released content by Red Hat on every invocation of `oc-mirror`. This way, it automatically picks up new releases and entirely new Operators. Constraints can be specified by only listing a desired set of Operators, which will not automatically add other newly released Operators into the mirror set. You can also specify a particular release channel, which limits mirroring to just this channel and not any new channels that have been added. This is important for Operator products, such as Red Hat Quay, that use different release channels for their minor releases. Lastly, you can specify a maximum version of a particular Operator, which causes the tool to only mirror the specified version range so that you do not automatically get any newer releases past the maximum version mirrored. In all these cases, you must update the image set configuration file to broaden the scope of the mirroring of Operators to get other Operators, new channels, and newer versions of Operators to be available in your target registry.

It is recommended to align constraints like channel specification or version ranges with the release strategy that a particular Operator has chosen. For example, when the Operator uses a **stable** channel, you should restrict mirroring to that channel and potentially a minimum version to find the right balance between download volume and getting stable updates regularly. If the Operator chooses a release version channel scheme, for example **stable-3.7**, you should mirror all releases in that channel. This allows you to keep receiving patch versions of the Operator, for example **3.7.1**. You can also regularly adjust the image set configuration to add channels for new product releases, for example **stable-3.8**.

Pruning images

Images are pruned automatically from the target mirror registry if they are no longer included in the latest image set that was generated and mirrored. This allows you to easily manage and clean up unneeded content and reclaim storage resources.

If there are OpenShift Container Platform releases or Operator versions that you no longer need, you can modify your image set configuration to exclude them, and they will be pruned from the mirror registry upon mirroring. This can be done by adjusting a minimum or maximum version range setting per

Operator in the image set configuration file or by deleting the Operator from the list of Operators to mirror from the catalog. You can also remove entire Operator catalogs or entire OpenShift Container Platform releases from the configuration file.



IMPORTANT

Images are not automatically pruned from the target mirror registry in the following situations:

- If there are no new or updated images to mirror
- If you are using the Technology Preview OCI feature

Additionally, if an Operator publisher removes an Operator version from a channel, the removed versions are pruned from the target mirror registry.

To disable automatic pruning of images from the target mirror registry, pass the **--skip-pruning** flag to the **oc mirror** command.

4.4.9.2. Updating your mirror registry content

After you publish the initial image set to the mirror registry, you can use the `oc-mirror` plugin to keep your disconnected clusters updated.

Depending on your image set configuration, `oc-mirror` automatically detects newer releases of OpenShift Container Platform and your selected Operators that have been released after you completed the initial mirror. It is recommended to run `oc-mirror` at regular intervals, for example in a nightly cron job, to receive product and security updates on a timely basis.

Prerequisites

- You have used the `oc-mirror` plugin to mirror the initial image set to your mirror registry.
- You have access to the storage backend that was used for the initial execution of the `oc-mirror` plugin.



NOTE

You must use the same storage backend as the initial execution of `oc-mirror` for the same mirror registry. Do not delete or modify the metadata image that is generated by the `oc-mirror` plugin.

Procedure

1. If necessary, update your image set configuration file to pick up new OpenShift Container Platform and Operator versions. See *Image set configuration examples* for example mirroring use cases.
2. Follow the same steps that you used to mirror your initial image set to the mirror registry. For instructions, see *Mirroring an image set in a partially disconnected environment* or *Mirroring an image set in a fully disconnected environment*.



IMPORTANT

- You must provide the same storage backend so that only a differential image set is created and mirrored.
- If you specified a top-level namespace for the mirror registry during the initial image set creation, then you must use this same namespace every time you run the `oc-mirror` plugin for the same mirror registry.

3. Configure your cluster to use the resources generated by `oc-mirror`.

Additional resources

- [Image set configuration examples](#)
- [Mirroring an image set in a partially disconnected environment](#)
- [Mirroring an image set in a fully disconnected environment](#)
- [Configuring your cluster to use the resources generated by `oc-mirror`](#)

4.4.10. Performing a dry run

You can use `oc-mirror` to perform a dry run, without actually mirroring any images. This allows you to review the list of images that would be mirrored, as well as any images that would be pruned from the mirror registry. It also allows you to catch any errors with your image set configuration early or use the generated list of images with other tools to carry out the mirroring operation.

Prerequisites

- You have access to the internet to obtain the necessary container images.
- You have installed the OpenShift CLI (**oc**).
- You have installed the **oc-mirror** CLI plugin.
- You have created the image set configuration file.

Procedure

1. Run the **oc mirror** command with the **--dry-run** flag to perform a dry run:

```
$ oc mirror --config=./imageset-config.yaml \ 1
docker://registry.example:5000 \ 2
--dry-run 3
```

- 1 Pass in the image set configuration file that was created. This procedure assumes that it is named **imageset-config.yaml**.
- 2 Specify the mirror registry. Nothing is mirrored to this registry as long as you use the **--dry-run** flag.
- 3 Use the **--dry-run** flag to generate the dry run artifacts and not an actual image set file.

Example output

```

Checking push permissions for registry.example:5000
Creating directory: oc-mirror-workspace/src/publish
Creating directory: oc-mirror-workspace/src/v2
Creating directory: oc-mirror-workspace/src/charts
Creating directory: oc-mirror-workspace/src/release-signatures
No metadata detected, creating new workspace
wrote mirroring manifests to oc-mirror-workspace/operators.1658342351/manifests-redhat-operator-index

...

info: Planning completed in 31.48s
info: Dry run complete
Writing image mapping to oc-mirror-workspace/mapping.txt

```

2. Navigate into the workspace directory that was generated:

```
$ cd oc-mirror-workspace/
```

3. Review the **mapping.txt** file that was generated.
This file contains a list of all images that would be mirrored.
4. Review the **pruning-plan.json** file that was generated.
This file contains a list of all images that would be pruned from the mirror registry when the image set is published.



NOTE

The **pruning-plan.json** file is only generated if your `oc-mirror` command points to your mirror registry and there are images to be pruned.

4.4.11. Mirroring file-based catalog Operator images in OCI format

You can use the `oc-mirror` plugin to mirror Operators in the Open Container Initiative (OCI) image format, instead of Docker v2 format. You can copy Operator images to a file-based catalog on disk in OCI format. Then you can copy local OCI images to your target mirror registry.



IMPORTANT

Using the `oc-mirror` plugin to mirror Operator images in OCI format is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

When using the OCI feature, images are not automatically pruned from the target mirror registry.

Prerequisites

Prerequisites

- You have access to the internet to obtain the necessary container images.
- You have installed the OpenShift CLI (**oc**).
- You have installed the **oc-mirror** CLI plugin.

Procedure

1. Optional: Retrieve the catalogs and images that you require and save them to disk. If you already have the catalog image in OCI format on disk, you can skip this step.
 - a. Create the image set configuration file:

Example image set configuration file for copying to disk

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
mirror:
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
  packages:
  - name: aws-load-balancer-operator
```



NOTE

When using the OCI feature, only the **mirror.operators.catalog** setting is available for use.

The **storageConfig** setting is ignored in favor of the location passed in to the **oc mirror** command.

- b. Run the **oc mirror** command to mirror the images from the specified image set configuration to disk:

```
$ oc mirror --config=./imageset-config.yaml \ 1
--use-oci-feature \ 2
--oci-feature-action=copy \ 3
oci://my-oci-catalog 4
```

- 1 Pass in the image set configuration file. This procedure assumes that it is named **imageset-config.yaml**.
- 2 Use the **--use-oci-feature** flag to enable the OCI feature.
- 3 To copy the catalog to disk, set the **--oci-feature-action** flag to **copy**.
- 4 Specify a directory on disk where you want to output the catalog. This procedure assumes that it is named **my-oci-catalog**. The path must start with **oci://**. If the specified directory is not a full path, the directory is created in the current working directory where the **oc mirror** command is run.



NOTE

You can optionally use the `--oci-registries-config` flag to specify the path to a TOML-formatted `registries.conf` file. You can use this to mirror from a different registry, such as a pre-production location for testing, without having to change the image set configuration file.

Example registries.conf file

```
[[registry]]
  location = "registry.redhat.io:5000"
  insecure = false
  blocked = false
  mirror-by-digest-only = true
  prefix = ""
[[registry.mirror]]
  location = "preprod-registry.example.com"
  insecure = false
```

Set the **location** field in the **registry.mirror** section to an alternative registry location that you want to pull images from. The **location** field in the **registry** section must be the same registry location as the one you specify in the image set configuration file.

- c. List your directory contents and verify that the following directories were created:

```
$ ls -l
```

Example output

```
my-oci-catalog      1
oc-mirror-workspace 2
olm_artifacts      3
```

- 1 Directory that contains the OCI catalog. This procedure assumes that it is named **my-oci-catalog**.
- 2 Directory that contains each image in the catalog in its original format.
- 3 Directory that contains the files that describe the Operator bundles that this catalog references.

2. Update the image set configuration file to specify the location of the catalog on disk to mirror to the target mirror registry:

Example image set configuration file for mirroring to mirror registry

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
mirror:
  operators:
```

```
- catalog: oci:///home/user/oc-mirror/my-oci-catalog/redhat-operator-index 1
packages:
- name: aws-load-balancer-operator
```

- 1** Specify the absolute path to the location of the OCI catalog on disk. This procedure assumes that you used **my-oci-catalog** as the directory and mirrored the **redhat-operator-index** catalog. The path must start with **oci://**.

- Run the `oc mirror` command to process the image set file on disk and mirror the contents to a target mirror registry:

```
$ oc mirror --config=./imageset-config.yaml \ 1
--use-oci-feature \ 2
--oci-feature-action=mirror \ 3
docker://registry.example:5000 4
```

- 1** Pass in the updated image set configuration file. This procedure assumes that it is named **imageset-config.yaml**.
- 2** Use the **--use-oci-feature** flag to enable the OCI feature.
- 3** To mirror the catalog to the target mirror registry, set the **--oci-feature-action** flag to **mirror**.
- 4** Specify the registry to mirror the image set file to. The registry must start with **docker://**. If you specify a top-level namespace for the mirror registry, you must also use this same namespace on subsequent executions.



NOTE

You can optionally use the **--oci-insecure-signature-policy** flag to not push signatures to the target mirror registry.

Next steps

- Configure your cluster to use the resources generated by `oc-mirror`.

Additional resources

- [File-based catalogs](#)

4.4.12. Image set configuration parameters

The `oc-mirror` plugin requires an image set configuration file that defines what images to mirror. The following table lists the available parameters for the **ImageSetConfiguration** resource.

Table 4.1. **ImageSetConfiguration** parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the ImageSetConfiguration content.	String. For example: mirror.openshift.io/v1alpha2 .
archiveSize	The maximum size, in GiB, of each archive file within the image set.	Integer. For example: 4
mirror	The configuration of the image set.	Object
mirror.additionalImages	The additional images configuration of the image set.	Array of objects. For example: <pre> additionalImages: - name: registry.redhat.io/ubi8/ubi:latest </pre>
mirror.additionalImages.name	The tag or digest of the image to mirror.	String. For example: registry.redhat.io/ubi8/ubi:latest
mirror.blockedImages	The full tag, digest, or pattern of images to block from mirroring.	Array of strings. For example: docker.io/library/alpine
mirror.helm	The helm configuration of the image set. Note that the oc-mirror plugin supports only helm charts that do not require user input when rendered.	Object
mirror.helm.local	The local helm charts to mirror.	Array of objects. For example: <pre> local: - name: podinfo path: /test/podinfo-5.0.0.tar.gz </pre>

Parameter	Description	Values
mirror.helm.local.name	The name of the local helm chart to mirror.	String. For example: podinfo .
mirror.helm.local.path	The path of the local helm chart to mirror.	String. For example: /test/podinfo-5.0.0.tar.gz .
mirror.helm.repositories	The remote helm repositories to mirror from.	Array of objects. For example: <pre> repositories: - name: podinfo url: https://example.github.io/podinfo charts: - name: podinfo version: 5.0.0 </pre>
mirror.helm.repositories.name	The name of the helm repository to mirror from.	String. For example: podinfo .
mirror.helm.repositories.url	The URL of the helm repository to mirror from.	String. For example: https://example.github.io/podinfo .
mirror.helm.repositories.charts	The remote helm charts to mirror.	Array of objects.
mirror.helm.repositories.charts.name	The name of the helm chart to mirror.	String. For example: podinfo .
mirror.helm.repositories.charts.version	The version of the named helm chart to mirror.	String. For example: 5.0.0 .

Parameter	Description	Values
mirror.operators	The Operators configuration of the image set.	Array of objects. For example: <pre>operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12 packages: - name: elasticsearch-operator minVersion: '2.4.0'</pre>
mirror.operators.catalog	The Operator catalog to include in the image set.	String. For example: registry.redhat.io/redhat/redhat-operator-index:v4.12.
mirror.operators.full	When true , downloads the full catalog, Operator package, or Operator channel.	Boolean. The default value is false .
mirror.operators.packages	The Operator packages configuration.	Array of objects. For example: <pre>operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12 packages: - name: elasticsearch-operator minVersion: '5.2.3-31'</pre>

Parameter	Description	Values
mirror.operators.packages.name	The Operator package name to include in the image set	String. For example: elasticsearch-operator .
mirror.operators.packages.channels	The Operator package channel configuration.	Object
mirror.operators.packages.channels.name	The Operator channel name, unique within a package, to include in the image set.	String. For example: fast or stable-v4.12 .
mirror.operators.packages.channels.maxVersion	The highest version of the Operator mirror across all channels in which it exists. See the following note for further information.	String. For example: 5.2.3-31
mirror.operators.packages.channels.minBundle	The name of the minimum bundle to include, plus all bundles in the upgrade graph to the channel head. Set this field only if the named bundle has no semantic version metadata.	String. For example: bundleName
mirror.operators.packages.channels.minVersion	The lowest version of the Operator to mirror across all channels in which it exists. See the following note for further information.	String. For example: 5.2.3-31
mirror.operators.packages.maxVersion	The highest version of the Operator to mirror across all channels in which it exists. See the following note for further information.	String. For example: 5.2.3-31 .
mirror.operators.packages.minVersion	The lowest version of the Operator to mirror across all channels in which it exists. See the following note for further information.	String. For example: 5.2.3-31 .
mirror.operators.skipDependencies	If true , dependencies of bundles are not included.	Boolean. The default value is false .
mirror.operators.targetName	Optional alternative name to mirror the referenced catalog as.	String. For example: my-operator-catalog

Parameter	Description	Values
mirror.operators.targetTag	Optional alternative tag to append to the targetName .	String. For example: v1
mirror.platform	The platform configuration of the image set.	Object
mirror.platform.architectures	The architecture of the platform release payload to mirror.	Array of strings. For example: <pre>architectures: - amd64 - arm64</pre>
mirror.platform.channels	The platform channel configuration of the image set.	Array of objects. For example: <pre>channels: - name: stable-4.10 - name: stable-4.12</pre>
mirror.platform.channels.full	When true , sets the minVersion to the first release in the channel and the maxVersion to the last release in the channel.	Boolean. The default value is false .
mirror.platform.channels.name	The name of the release channel.	String. For example: stable-4.12
mirror.platform.channels.minVersion	The minimum version of the referenced platform to be mirrored.	String. For example: 4.9.6
mirror.platform.channels.maxVersion	The highest version of the referenced platform to be mirrored.	String. For example: 4.12.1
mirror.platform.channels.shortestPath	Toggles shortest path mirroring or full range mirroring.	Boolean. The default value is false .
mirror.platform.channels.type	The type of the platform to be mirrored.	String. For example: ocp or okd . The default is ocp .

Parameter	Description	Values
mirror.platform.graph	Indicates whether the OSUS graph is added to the image set and subsequently published to the mirror.	Boolean. The default value is false .
storageConfig	The back-end configuration of the image set.	Object
storageConfig.local	The local back-end configuration of the image set.	Object
storageConfig.local.path	The path of the directory to contain the image set metadata.	String. For example: ./path/to/dir/ .
storageConfig.registry	The registry back-end configuration of the image set.	Object
storageConfig.registry.imageURL	The back-end registry URI. Can optionally include a namespace reference in the URI.	String. For example: quay.io/myuser/imageset:metadata .
storageConfig.registry.skipTLS	Optionally skip TLS verification of the referenced back-end registry.	Boolean. The default value is false .

NOTE

Using the the **minVersion** and **maxVersion** properties to filter for a specific Operator version range can result in a multiple channel heads error. The error message will state that there are **multiple channel heads**. This is because when the filter is applied, the update graph of the operator is truncated.

The Operator Lifecycle Manager requires that every operator channel contains versions that form an update graph with exactly one end point, that is, the latest version of the operator. When applying the filter range that graph can turn into two or more separate graphs or a graph that has more than one end point.

To avoid this error, do not filter out the latest version of an operator. If you still run into the error, depending on the operator, either the **maxVersion** property needs to be increased or the **minVersion** property needs to be decreased. Because every operator graph can be different, you might need to adjust these values, according to the procedure, until the error is gone.

4.4.13. Image set configuration examples

The following **ImageSetConfiguration** file examples show the configuration for various mirroring use cases.

Use case: Including the shortest OpenShift Container Platform upgrade path

The following **ImageSetConfiguration** file uses a local storage backend and includes all OpenShift Container Platform versions along the shortest upgrade path from the minimum version of **4.11.37** to the maximum version of **4.12.15**.

Example ImageSetConfiguration file

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
      - name: stable-4.12
        minVersion: 4.11.37
        maxVersion: 4.12.15
        shortestPath: true
```

Use case: Including all versions of OpenShift Container Platform from a minimum to the latest

The following **ImageSetConfiguration** file uses a registry storage backend and includes all OpenShift Container Platform versions starting at a minimum version of **4.10.10** to the latest version in the channel.

On every invocation of oc-mirror with this image set configuration, the latest release of the **stable-4.10** channel is evaluated, so running oc-mirror at regular intervals ensures that you automatically receive the latest releases of OpenShift Container Platform images.

Example ImageSetConfiguration file

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  platform:
    channels:
      - name: stable-4.10
        minVersion: 4.10.10
```

Use case: Including Operator versions from a minimum to the latest

The following **ImageSetConfiguration** file uses a local storage backend and includes only the Red Hat Advanced Cluster Security for Kubernetes Operator, versions starting at 4.0.1 and later in the **stable** channel.

**NOTE**

When you specify a minimum or maximum version range, you might not receive all Operator versions in that range.

By default, `oc-mirror` excludes any versions that are skipped or replaced by a newer version in the Operator Lifecycle Manager (OLM) specification. Operator versions that are skipped might be affected by a CVE or contain bugs. Use a newer version instead. For more information on skipped and replaced versions, see [Creating an update graph with OLM](#).

To receive all Operator versions in a specified range, you can set the `mirror.operators.full` field to `true`.

Example ImageSetConfiguration file

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
      packages:
        - name: rhacs-operator
          channels:
            - name: stable
            minVersion: 4.0.1
```

**NOTE**

To specify a maximum version instead of the latest, set the `mirror.operators.packages.channels.maxVersion` field.

Use case: Including the Nutanix CSI Operator

The following **ImageSetConfiguration** file uses a local storage backend and includes the Nutanix CSI Operator, the OpenShift Update Service (OSUS) graph image, and an additional Red Hat Universal Base Image (UBI).

Example ImageSetConfiguration file

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  registry:
    imageURL: mylocalregistry/ocp-mirror/openshift4
    skipTLS: false
mirror:
  platform:
    channels:
      - name: stable-4.12
        type: ocp
    graph: true
```

```

operators:
- catalog: registry.redhat.io/redhat/certified-operator-index:v4.12
  packages:
  - name: nutanixcsioperator
    channels:
    - name: stable
additionalImages:
- name: registry.redhat.io/ubi9/ubi:latest

```

Use case: Including the default Operator channel

The following **ImageSetConfiguration** file includes the **stable-5.7** and **stable** channels for the OpenShift Elasticsearch Operator. Even if only the packages from the **stable-5.7** channel are needed, the **stable** channel must also be included in the **ImageSetConfiguration** file, because it is the default channel for the Operator. You must always include the default channel for the Operator package even if you do not use the bundles in that channel.

TIP

You can find the default channel by running the following command: **oc mirror list operators --catalog=<catalog_name> --package=<package_name>**.

Example ImageSetConfiguration file

```

apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
    packages:
    - name: elasticsearch-operator
      channels:
      - name: stable-5.7
      - name: stable

```

Use case: Including an entire catalog (all versions)

The following **ImageSetConfiguration** file sets the **mirror.operators.full** field to **true** to include all versions for an entire Operator catalog.

Example ImageSetConfiguration file

```

apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
    full: true

```

Use case: Including an entire catalog (channel heads only)

The following **ImageSetConfiguration** file includes the channel heads for an entire Operator catalog.

By default, for each Operator in the catalog, `oc-mirror` includes the latest Operator version (channel head) from the default channel. If you want to mirror all Operator versions, and not just the channel heads, you must set the **mirror.operators.full** field to **true**.

Example ImageSetConfiguration file

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
```

Use case: Including arbitrary images and helm charts

The following **ImageSetConfiguration** file uses a registry storage backend and includes helm charts and an additional Red Hat Universal Base Image (UBI).

Example ImageSetConfiguration file

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
archiveSize: 4
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  platform:
    architectures:
      - "s390x"
    channels:
      - name: stable-4.12
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
  helm:
    repositories:
      - name: redhat-helm-charts
        url: https://raw.githubusercontent.com/redhat-developer/redhat-helm-charts/master
    charts:
      - name: ibm-mongodb-enterprise-helm
        version: 0.2.0
  additionalImages:
    - name: registry.redhat.io/ubi9/ubi:latest
```

4.4.14. Command reference for oc-mirror

The following tables describe the **oc mirror** subcommands and flags:

Table 4.2. oc mirror subcommands

Subcommand	Description
completion	Generate the autocompletion script for the specified shell.
describe	Output the contents of an image set.
help	Show help about any subcommand.
init	Output an initial image set configuration template.
list	List available platform and Operator content and their version.
version	Output the oc-mirror version.

Table 4.3. oc mirror flags

Flag	Description
-c, --config <string>	Specify the path to an image set configuration file.
--continue-on-error	If any non image-pull related error occurs, continue and attempt to mirror as much as possible.
--dest-skip-tls	Disable TLS validation for the target registry.
--dest-use-http	Use plain HTTP for the target registry.
--dry-run	Print actions without mirroring images. Generates mapping.txt and pruning-plan.json files.
--from <string>	Specify the path to an image set archive that was generated by an execution of oc-mirror to load into a target registry.
-h, --help	Show the help.
--ignore-history	Ignore past mirrors when downloading images and packing layers. Disables incremental mirroring and might download more data.
--manifests-only	Generate manifests for ImageContentSourcePolicy objects to configure a cluster to use the mirror registry, but do not actually mirror any images. To use this flag, you must pass in an image set archive with the --from flag.
--max-nested-paths <int>	Specify the maximum number of nested paths for destination registries that limit nested paths. The default is 2 .
--max-per-registry <int>	Specify the number of concurrent requests allowed per registry. The default is 6 .

Flag	Description
--oci-feature-action	The action to perform when using the Technology Preview OCI feature. The options are copy or mirror .
--oci-insecure-signature-policy	Do not push signatures when using the Technology Preview OCI feature.
--oci-registries-config	Provide a registries configuration file to specify an alternative registry location to copy from when using the Technology Preview OCI feature.
--skip-cleanup	Skip removal of artifact directories.
--skip-image-pin	Do not replace image tags with digest pins in Operator catalogs.
--skip-metadata-check	Skip metadata when publishing an image set. This is only recommended when the image set was created with --ignore-history .
--skip-missing	If an image is not found, skip it instead of reporting an error and aborting execution. Does not apply to custom images explicitly specified in the image set configuration.
--skip-pruning	Disable automatic pruning of images from the target mirror registry.
--skip-verification	Skip digest verification.
--source-skip-tls	Disable TLS validation for the source registry.
--source-use-http	Use plain HTTP for the source registry.
--use-oci-feature	Use the Technology Preview OCI feature for copying OCI-formatted images.
-v, --verbose <int>	Specify the number for the log level verbosity. Valid values are 0 - 9 . The default is 0 .

4.4.15. Additional resources

- [About cluster updates in a disconnected environment](#)

CHAPTER 5. INSTALLING ON ALIBABA

5.1. PREPARING TO INSTALL ON ALIBABA CLOUD



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

5.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

5.1.2. Requirements for installing OpenShift Container Platform on Alibaba Cloud

Before installing OpenShift Container Platform on Alibaba Cloud, you must configure and register your domain, create a Resource Access Management (RAM) user for the installation, and review the supported Alibaba Cloud data center regions and zones for the installation.

5.1.3. Registering and Configuring Alibaba Cloud Domain

To install OpenShift Container Platform, the Alibaba Cloud account you use must have a dedicated public hosted zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Alibaba Cloud or another source.



NOTE

If you purchase a new domain through Alibaba Cloud, it takes time for the relevant DNS changes to propagate. For more information about purchasing domains through Alibaba Cloud, see [Alibaba Cloud domains](#).

2. If you are using an existing domain and registrar, migrate its DNS to Alibaba Cloud. See [Domain name transfer](#) in the Alibaba Cloud documentation.
3. Configure DNS for your domain. This includes:
 - [Registering a generic domain name](#).

- [Completing real-name verification for your domain name](#) .
 - [Applying for an Internet Content Provider \(ICP\) filing](#) .
 - [Enabling domain name resolution](#).
Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
4. If you are using a subdomain, follow the procedures of your company to add its delegation records to the parent domain.

5.1.4. Supported Alibaba regions

You can deploy an OpenShift Container Platform cluster to the regions listed in the [Alibaba Regions and zones documentation](#).

5.1.5. Next steps

- [Create the required Alibaba Cloud resources](#) .

5.2. CREATING THE REQUIRED ALIBABA CLOUD RESOURCES

Before you install OpenShift Container Platform, you must use the Alibaba Cloud console to create a Resource Access Management (RAM) user that has sufficient permissions to install OpenShift Container Platform into your Alibaba Cloud. This user must also have permissions to create new RAM users. You can also configure and use the **ccocli** tool to create new credentials for the OpenShift Container Platform components with the permissions that they require.



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

5.2.1. Creating the required RAM user

You must have a Alibaba Cloud Resource Access Management (RAM) user for the installation that has sufficient privileges. You can use the Alibaba Cloud Resource Access Management console to create a new user or modify an existing user. Later, you create credentials in OpenShift Container Platform based on this user's permissions.

When you configure the RAM user, be sure to consider the following requirements:

- The user must have an Alibaba Cloud AccessKey ID and AccessKey secret pair.
 - For a new user, you can select **Open API Access** for the Access Mode when creating the user. This mode generates the required AccessKey pair.

- For an existing user, you can add an AccessKey pair or you can [obtain the AccessKey pair](#) for that user.



NOTE

When created, the AccessKey secret is displayed only once. You must immediately save the AccessKey pair because the AccessKey pair is required for API calls.

- Add the AccessKey ID and secret to the `~/.alibabacloud/credentials` file on your local computer. Alibaba Cloud automatically creates this file when you log in to the console. The Cloud Credential Operator (CCO) utility, `ccoutil`, uses these credentials when processing **Credential Request** objects.

For example:

```
[default]                # Default client
type = access_key        # Certification type: access_key
access_key_id = LTAI5t8cefXKmt    # Key 1
access_key_secret = wYx56mszAN4Uunfh    # Secret
```

- Add your AccessKeyID and AccessKeySecret here.

- The RAM user must have the **AdministratorAccess** policy to ensure that the account has sufficient permission to create the OpenShift Container Platform cluster. This policy grants permissions to manage all Alibaba Cloud resources. When you attach the **AdministratorAccess** policy to a RAM user, you grant that user full access to all Alibaba Cloud services and resources. If you do not want to create a user with full access, create a custom policy with the following actions that you can add to your RAM user for installation. These actions are sufficient to install OpenShift Container Platform.

TIP

You can copy and paste the following JSON code into the Alibaba Cloud console to create a custom policy. For information on creating custom policies, see [Create a custom policy](#) in the Alibaba Cloud documentation.

Example 5.1. Example custom policy JSON file

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "tag:ListTagResources",
        "tag:UntagResources"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "vpc:DescribeVpcs",
        "vpc:DeleteVpc",

```

```

    "vpc:DescribeVSwitches",
    "vpc:DeleteVSwitch",
    "vpc:DescribeEipAddresses",
    "vpc:DescribeNatGateways",
    "vpc:ReleaseEipAddress",
    "vpc:DeleteNatGateway",
    "vpc:DescribeSnatTableEntries",
    "vpc:CreateSnatEntry",
    "vpc:AssociateEipAddress",
    "vpc:ListTagResources",
    "vpc:TagResources",
    "vpc:DescribeVSwitchAttributes",
    "vpc:CreateVSwitch",
    "vpc:CreateNatGateway",
    "vpc:DescribeRouteTableList",
    "vpc:CreateVpc",
    "vpc:AllocateEipAddress",
    "vpc:ListEnhancedNatGatewayAvailableZones"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "ecs:ModifyInstanceAttribute",
    "ecs:DescribeSecurityGroups",
    "ecs>DeleteSecurityGroup",
    "ecs:DescribeSecurityGroupReferences",
    "ecs:DescribeSecurityGroupAttribute",
    "ecs:RevokeSecurityGroup",
    "ecs:DescribeInstances",
    "ecs>DeleteInstances",
    "ecs:DescribeNetworkInterfaces",
    "ecs:DescribeInstanceRamRole",
    "ecs:DescribeUserData",
    "ecs:DescribeDisks",
    "ecs:ListTagResources",
    "ecs:AuthorizeSecurityGroup",
    "ecs:RunInstances",
    "ecs:TagResources",
    "ecs:ModifySecurityGroupPolicy",
    "ecs:CreateSecurityGroup",
    "ecs:DescribeAvailableResource",
    "ecs:DescribeRegions",
    "ecs:AttachInstanceRamRole"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "pvtz:DescribeRegions",
    "pvtz:DescribeZones",
    "pvtz>DeleteZone",
    "pvtz>DeleteZoneRecord",
    "pvtz:BindZoneVpc",

```

```

    "pvtz:DescribeZoneRecords",
    "pvtz:AddZoneRecord",
    "pvtz:SetZoneRecordStatus",
    "pvtz:DescribeZoneInfo",
    "pvtz:DescribeSyncEcsHostTask",
    "pvtz:AddZone"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "slb:DescribeLoadBalancers",
    "slb:SetLoadBalancerDeleteProtection",
    "slb>DeleteLoadBalancer",
    "slb:SetLoadBalancerModificationProtection",
    "slb:DescribeLoadBalancerAttribute",
    "slb:AddBackendServers",
    "slb:DescribeLoadBalancerTCPLListenerAttribute",
    "slb:SetLoadBalancerTCPLListenerAttribute",
    "slb:StartLoadBalancerListener",
    "slb:CreateLoadBalancerTCPLListener",
    "slb:ListTagResources",
    "slb:TagResources",
    "slb:CreateLoadBalancer"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "ram:ListResourceGroups",
    "ram>DeleteResourceGroup",
    "ram:ListPolicyAttachments",
    "ram:DetachPolicy",
    "ram:GetResourceGroup",
    "ram>CreateResourceGroup",
    "ram>DeleteRole",
    "ram:GetPolicy",
    "ram>DeletePolicy",
    "ram:ListPoliciesForRole",
    "ram>CreateRole",
    "ram:AttachPolicyToRole",
    "ram:GetRole",
    "ram>CreatePolicy",
    "ram>CreateUser",
    "ram:DetachPolicyFromRole",
    "ram>CreatePolicyVersion",
    "ram:DetachPolicyFromUser",
    "ram:ListPoliciesForUser",
    "ram:AttachPolicyToUser",
    "ram>CreateUser",
    "ram:GetUser",
    "ram>DeleteUser",
    "ram>CreateAccessKey",
    "ram:ListAccessKeys",

```

```

    "ram:DeleteAccessKey",
    "ram:ListUsers",
    "ram:ListPolicyVersions"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "oss:DeleteBucket",
    "oss:DeleteBucketTagging",
    "oss:GetBucketTagging",
    "oss:GetBucketCors",
    "oss:GetBucketPolicy",
    "oss:GetBucketLifecycle",
    "oss:GetBucketReferer",
    "oss:GetBucketTransferAcceleration",
    "oss:GetBucketLog",
    "oss:GetBucketWebSite",
    "oss:GetBucketInfo",
    "oss:PutBucketTagging",
    "oss:PutBucket",
    "oss:OpenOssService",
    "oss:ListBuckets",
    "oss:GetService",
    "oss:PutBucketACL",
    "oss:GetBucketLogging",
    "oss:ListObjects",
    "oss:GetObject",
    "oss:PutObject",
    "oss:DeleteObject"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "alidns:DescribeDomainRecords",
    "alidns>DeleteDomainRecord",
    "alidns:DescribeDomains",
    "alidns:DescribeDomainRecordInfo",
    "alidns:AddDomainRecord",
    "alidns:SetDomainRecordStatus"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": "bssapi>CreateInstance",
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": "ram:PassRole",
  "Resource": "*",
  "Effect": "Allow",

```

```

"Condition": {
  "StringEquals": {
    "acs:Service": "ecs.aliyuncs.com"
  }
}
]
}

```

For more information about creating a RAM user and granting permissions, see [Create a RAM user](#) and [Grant permissions to a RAM user](#) in the Alibaba Cloud documentation.

5.2.2. Configuring the Cloud Credential Operator utility

To assign RAM users and policies that provide long-lived RAM AccessKeys (AKs) for each in-cluster component, extract and prepare the Cloud Credential Operator (CCO) utility (**ccoctl**) binary.



NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



NOTE

Ensure that the architecture of the **\$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE --file="/usr/bin/ccoctl" -a ~/.pull-secret
```

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl
```

Verification

- To verify that **ccoctl** is ready to use, display the help file by running the following command:

```
$ ccoctl --help
```

Output of **ccoctl --help**

```
OpenShift credentials provisioning tool
```

```
Usage:
```

```
ccoctl [command]
```

```
Available Commands:
```

```
alibabacloud Manage credentials objects for alibaba cloud
```

```
aws          Manage credentials objects for AWS cloud
```

```
gcp          Manage credentials objects for Google cloud
```

```
help        Help about any command
```

```
ibmcloud    Manage credentials objects for IBM Cloud
```

```
nutanix     Manage credentials objects for Nutanix
```

```
Flags:
```

```
-h, --help  help for ccoctl
```

```
Use "ccoctl [command] --help" for more information about a command.
```

Additional resources

- [Preparing to update a cluster with manually maintained credentials](#)

5.2.3. Next steps

- Install a cluster on Alibaba Cloud infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:
 - [Installing a cluster quickly on Alibaba Cloud](#) You can install a cluster quickly by using the default configuration options.
 - [Installing a customized cluster on Alibaba Cloud](#) The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).

5.3. INSTALLING A CLUSTER QUICKLY ON ALIBABA CLOUD

In OpenShift Container Platform version 4.12, you can install a cluster on Alibaba Cloud that uses the default configuration options.



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

5.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [registered your domain](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You have [created the required Alibaba Cloud resources](#).
- If the cloud Resource Access Management (RAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the kube-system namespace, you can [manually create and maintain Resource Access Management \(RAM\) credentials](#).

5.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

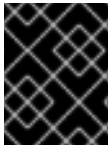
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

5.3.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

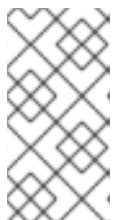
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

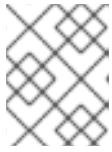
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

5.3.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.3.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Alibaba Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

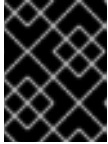
- ii. Select **alibabacloud** as the platform to target.
 - iii. Select the region to deploy the cluster to.
 - iv. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - v. Provide a descriptive name for your cluster.
 - vi. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Installing the cluster into Alibaba Cloud requires that the Cloud Credential Operator (CCO) operate in manual mode. Modify the **install-config.yaml** file to set the **credentialsMode** parameter to **Manual**:

Example `install-config.yaml` configuration file with **credentialsMode** set to **Manual**

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1 Add this line to set the **credentialsMode** to **Manual**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

5.3.6. Generating the required installation manifests

You must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

Procedure

1. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the directory in which the installation program creates files.

5.3.7. Creating credentials for OpenShift Container Platform components with the **ccoctl** tool

You can use the OpenShift Container Platform Cloud Credential Operator (CCO) utility to automate the creation of Alibaba Cloud RAM users and policies for each in-cluster component.



NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Created a RAM user with sufficient permission to create the OpenShift Container Platform cluster.
- Added the AccessKeyID (**access_key_id**) and AccessKeySecret (**access_key_secret**) of that RAM user into the [~/alibabacloud/credentials](#) file on your local computer.

Procedure

1. Set the **\$RELEASE_IMAGE** variable by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
--credentials-requests \
--cloud=alibabacloud \
--to=<path_to_directory_with_list_of_credentials_requests>/credrequests \ 1
$RELEASE_IMAGE
```

- 1** **credrequests** is the directory where the list of **CredentialsRequest** objects is stored. This command creates the directory if it does not exist.



NOTE

This command can take a few moments to run.

3. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example **credrequests** directory contents for OpenShift Container Platform 4.12 on Alibaba Cloud

```
0000_30_machine-api-operator_00_credentials-request.yaml 1
0000_50_cluster-image-registry-operator_01-registry-credentials-request-alibaba.yaml 2
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 3
0000_50_cluster-storage-operator_03_credentials_request_alibaba.yaml 4
```

- 1** The Machine API Operator CR is required.
- 2** The Image Registry Operator CR is required.
- 3** The Ingress Operator CR is required.
- 4** The Storage Operator CR is an optional component and might be disabled in your cluster.

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects in the **credrequests** directory:
- a. Run the following command to use the tool:

```
$ ccoctl alibabacloud create-ram-users \
--name <name> \
--region=<alibaba_region> \
--credentials-requests-dir=
<path_to_directory_with_list_of_credentials_requests>/credrequests \
--output-dir=<path_to_ccoctl_output_dir>
```

where:

- **<name>** is the name used to tag any cloud resources that are created for tracking.
- **<alibaba_region>** is the Alibaba Cloud region in which cloud resources will be created.
- **<path_to_directory_with_list_of_credentials_requests>/credrequests** is the directory containing the files for the component **CredentialsRequest** objects.

- `<path_to_ccoctl_output_dir>` is the directory where the generated component credentials secrets will be placed.



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

Example output

```
2022/02/11 16:18:26 Created RAM User: user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:27 Ready for creating new ram policy user1-alicloud-openshift-
machine-api-alibabacloud-credentials-policy-policy
2022/02/11 16:18:27 RAM policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has created
2022/02/11 16:18:28 Policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has attached on user user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:29 Created access keys for RAM User: user1-alicloud-openshift-
machine-api-alibabacloud-credentials
2022/02/11 16:18:29 Saved credentials configuration to: user1-
alicloud/manifests/openshift-machine-api-alibabacloud-credentials-credentials.yaml
...
```



NOTE

A RAM user can have up to two AccessKeys at the same time. If you run **ccoctl alibabacloud create-ram-users** more than twice, the previous generated manifests secret becomes stale and you must reapply the newly generated secrets.

- Verify that the OpenShift Container Platform secrets are created:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output:

```
openshift-cluster-csi-drivers-alibaba-disk-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-alibabacloud-credentials-credentials.yaml
```

You can verify that the RAM users and policies are created by querying Alibaba Cloud. For more information, refer to Alibaba Cloud documentation on listing RAM users and policies.

- Copy the generated credential files to the target manifests directory:

```
$ cp ./<path_to_ccoctl_output_dir>/manifests/*credentials.yaml
./<path_to_installation>dir>/manifests/
```

where:

<path_to_ccoctl_output_dir>

Specifies the directory created by the **ccoctl alibabacloud create-ram-users** command.

<path_to_installation_dir>

Specifies the directory in which the installation program creates files.

5.3.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

5.3.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.

5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

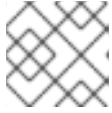
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

5.3.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

5.3.11. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https      reencrypt/Redirect      None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

5.3.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.
- See [About remote health monitoring](#) for more information about the Telemetry service

5.3.13. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

5.4. INSTALLING A CLUSTER ON ALIBABA CLOUD WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on infrastructure that the installation program provisions on Alibaba Cloud. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.



NOTE

The scope of the OpenShift Container Platform installation configurations is intentionally narrow. It is designed for simplicity and ensured success. You can complete many more OpenShift Container Platform configuration tasks after an installation completes.



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

5.4.1. Prerequisites

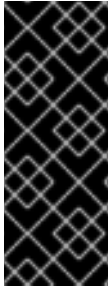
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [registered your domain](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud Resource Access Management (RAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain Resource Access Management \(RAM\) credentials](#).

5.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

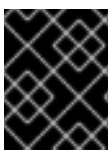
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

5.4.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

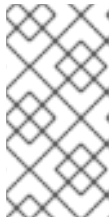
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

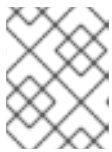
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

5.4.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.4.4.1. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Alibaba Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **alibabacloud** as the platform to target.
- iii. Select the region to deploy the cluster to.
- iv. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
- v. Provide a descriptive name for your cluster.
- vi. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .

- Installing the cluster into Alibaba Cloud requires that the Cloud Credential Operator (CCO) operate in manual mode. Modify the **install-config.yaml** file to set the **credentialsMode** parameter to **Manual**:

Example **install-config.yaml** configuration file with **credentialsMode** set to **Manual**

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1** Add this line to set the **credentialsMode** to **Manual**.

- Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
- Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

5.4.4.2. Generating the required installation manifests

You must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

Procedure

- Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the directory in which the installation program creates files.

5.4.4.3. Creating credentials for OpenShift Container Platform components with the **ccoctl** tool

You can use the OpenShift Container Platform Cloud Credential Operator (CCO) utility to automate the creation of Alibaba Cloud RAM users and policies for each in-cluster component.



NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Created a RAM user with sufficient permission to create the OpenShift Container Platform cluster.
- Added the AccessKeyID (**access_key_id**) and AccessKeySecret (**access_key_secret**) of that RAM user into the [~/alibabacloud/credentials](#) file on your local computer.

Procedure

1. Set the **\$RELEASE_IMAGE** variable by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
--credentials-requests \
--cloud=alibabacloud \
--to=<path_to_directory_with_list_of_credentials_requests>/credrequests \ 1
$RELEASE_IMAGE
```

- 1** **credrequests** is the directory where the list of **CredentialsRequest** objects is stored. This command creates the directory if it does not exist.



NOTE

This command can take a few moments to run.

3. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on Alibaba Cloud

```
0000_30_machine-api-operator_00_credentials-request.yaml 1
0000_50_cluster-image-registry-operator_01-registry-credentials-request-alibaba.yaml 2
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 3
0000_50_cluster-storage-operator_03_credentials_request_alibaba.yaml 4
```

- 1** The Machine API Operator CR is required.

- 2 The Image Registry Operator CR is required.
- 3 The Ingress Operator CR is required.
- 4 The Storage Operator CR is an optional component and might be disabled in your cluster.

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects in the **credrequests** directory:

- a. Run the following command to use the tool:

```
$ ccoctl alibabacloud create-ram-users \
--name <name> \
--region=<alibaba_region> \
--credentials-requests-dir=
<path_to_directory_with_list_of_credentials_requests>/credrequests \
--output-dir=<path_to_ccoctl_output_dir>
```

where:

- **<name>** is the name used to tag any cloud resources that are created for tracking.
- **<alibaba_region>** is the Alibaba Cloud region in which cloud resources will be created.
- **<path_to_directory_with_list_of_credentials_requests>/credrequests** is the directory containing the files for the component **CredentialsRequest** objects.
- **<path_to_ccoctl_output_dir>** is the directory where the generated component credentials secrets will be placed.



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

Example output

```
2022/02/11 16:18:26 Created RAM User: user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:27 Ready for creating new ram policy user1-alicloud-openshift-
machine-api-alibabacloud-credentials-policy-policy
2022/02/11 16:18:27 RAM policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has created
2022/02/11 16:18:28 Policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has attached on user user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:29 Created access keys for RAM User: user1-alicloud-openshift-
machine-api-alibabacloud-credentials
2022/02/11 16:18:29 Saved credentials configuration to: user1-
alicloud/manifests/openshift-machine-api-alibabacloud-credentials-credentials.yaml
...
```

**NOTE**

A RAM user can have up to two AccessKeys at the same time. If you run **ccoctl alibabacloud create-ram-users** more than twice, the previous generated manifests secret becomes stale and you must reapply the newly generated secrets.

- b. Verify that the OpenShift Container Platform secrets are created:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output:

```
openshift-cluster-csi-drivers-alibaba-disk-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-alibabacloud-credentials-credentials.yaml
```

You can verify that the RAM users and policies are created by querying Alibaba Cloud. For more information, refer to Alibaba Cloud documentation on listing RAM users and policies.

5. Copy the generated credential files to the target manifests directory:

```
$ cp ./<path_to_ccoctl_output_dir>/manifests/*credentials.yaml
./<path_to_installation>dir>/manifests/
```

where:

<path_to_ccoctl_output_dir>

Specifies the directory created by the **ccoctl alibabacloud create-ram-users** command.

<path_to_installation_dir>

Specifies the directory in which the installation program creates files.

5.4.4.4. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

5.4.4.4.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.1. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

5.4.4.4.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 5.2. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


5.4.4.4.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 5.3. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="486 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="486 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; width: 65px; height: 360px; margin-bottom: 10px;"></div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="488 1440 592 1637" style="background-color: black; width: 65px; height: 88px; margin-bottom: 10px;"></div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

5.4.4.4.4. Additional Alibaba Cloud configuration parameters

Additional Alibaba Cloud configuration parameters are described in the following table. The **alibabacloud** parameters are the configuration used when installing on Alibaba Cloud. The **defaultMachinePlatform** parameters are the default configuration used when installing on Alibaba Cloud for machine pools that do not define their own platform configuration.

These parameters apply to both compute machines and control plane machines where specified.



NOTE

If defined, the parameters **compute.platform.alibabacloud** and **controlPlane.platform.alibabacloud** will overwrite **platform.alibabacloud.defaultMachinePlatform** settings for compute machines and control plane machines respectively.

Table 5.4. Optional Alibaba Cloud parameters

Parameter	Description	Values
compute.platform.alibabacLOUD.imageID	The imageID used to create the ECS instance. ImageID must belong to the same region as the cluster.	String.
compute.platform.alibabacLOUD.instanceType	InstanceType defines the ECS instance type. Example: ecs.g6.large	String.
compute.platform.alibabacLOUD.systemDiskCategory	Defines the category of the system disk. Examples: cloud_efficiency,cloud_essd	String.
compute.platform.alibabacLOUD.systemDiskSize	Defines the size of the system disk in gibibytes (GiB).	Integer.
compute.platform.alibabacLOUD.zones	The list of availability zones that can be used. Examples: cn-hangzhou-h, cn-hangzhou-j	String list.
controlPlane.platform.alibabacLOUD.imageID	The imageID used to create the ECS instance. ImageID must belong to the same region as the cluster.	String.
controlPlane.platform.alibabacLOUD.instanceType	InstanceType defines the ECS instance type. Example: ecs.g6.xlarge	String.
controlPlane.platform.alibabacLOUD.systemDiskCategory	Defines the category of the system disk. Examples: cloud_efficiency,cloud_essd	String.
controlPlane.platform.alibabacLOUD.systemDiskSize	Defines the size of the system disk in gibibytes (GiB).	Integer.
controlPlane.platform.alibabacLOUD.zones	The list of availability zones that can be used. Examples: cn-hangzhou-h, cn-hangzhou-j	String list.

Parameter	Description	Values
platform.alibabacloud.region	Required. The Alibaba Cloud region where the cluster will be created.	String.
platform.alibabacloud.resourceGroupID	The ID of an already existing resource group where the cluster will be installed. If empty, the installation program will create a new resource group for the cluster.	String.
platform.alibabacloud.tags	Additional keys and values to apply to all Alibaba Cloud resources created for the cluster.	Object.
platform.alibabacloud.vpcID	The ID of an already existing VPC where the cluster should be installed. If empty, the installation program will create a new VPC for the cluster.	String.
platform.alibabacloud.vswitchIDs	The ID list of already existing VSwitches where cluster resources will be created. The existing VSwitches can only be used when also using existing VPC. If empty, the installation program will create new VSwitches for the cluster.	String list.
platform.alibabacloud.defaultMachinePlatformImageID	For both compute machines and control plane machines, the image ID that should be used to create ECS instance. If set, the image ID should belong to the same region as the cluster.	String.
platform.alibabacloud.defaultMachinePlatformInstanceType	For both compute machines and control plane machines, the ECS instance type used to create the ECS instance. Example: ecs.g6.xlarge	String.

Parameter	Description	Values
platform.alibabacloud.defaultMachinePlatform.systemDiskCategory	For both compute machines and control plane machines, the category of the system disk. Examples: cloud_efficiency , cloud_essd .	String, for example "", cloud_efficiency , cloud_essd .
platform.alibabacloud.defaultMachinePlatform.systemDiskSize	For both compute machines and control plane machines, the size of the system disk in gibibytes (GiB). The minimum is 120 .	Integer.
platform.alibabacloud.defaultMachinePlatform.zones	For both compute machines and control plane machines, the list of availability zones that can be used. Examples: cn-hangzhou-h , cn-hangzhou-j	String list.
platform.alibabacloud.privateZoneID	The ID of an existing private zone into which to add DNS records for the cluster's internal API. An existing private zone can only be used when also using existing VPC. The private zone must be associated with the VPC containing the subnets. Leave the private zone unset to have the installation program create the private zone on your behalf.	String.

5.4.4.5. Sample customized install-config.yaml file for Alibaba Cloud

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: alicloud-dev.devcluster.openshift.com
credentialsMode: Manual
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64

```

```

hyperthreading: Enabled
name: master
platform: {}
replicas: 3
metadata:
  creationTimestamp: null
  name: test-cluster ❶
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes ❷
  serviceNetwork:
  - 172.30.0.0/16
platform:
  alibabacloud:
    defaultMachinePlatform: ❸
    instanceType: ecs.g6.xlarge
    systemDiskCategory: cloud_efficiency
    systemDiskSize: 200
    region: ap-southeast-1 ❹
    resourceGroupID: rg-acfnw6j3hyai ❺
    vpcID: vpc-0xifdjerdibmaqvjob2b ❻
    vswitchIDs: ❼
    - vsw-0xi8ycgwc8wv5rhviwdq5
    - vsw-0xiy6v3z2tedv009b4pz2
  publish: External
  pullSecret: '{"auths": {"cloud.openshift.com": {"auth": ... }}' ❽
  sshKey: |
    ssh-rsa AAAA... ❾

```

- ❶ Required. The installation program prompts you for a cluster name.
- ❷ The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- ❸ Optional. Specify parameters for machine pools that do not define their own platform configuration.
- ❹ Required. The installation program prompts you for the region to deploy the cluster to.
- ❺ Optional. Specify an existing resource group where the cluster should be installed.
- ❽ Required. The installation program prompts you for the pull secret.
- ❾ Optional. The installation program prompts you for the SSH key value that you use to access the machines in your cluster.
- ❻�� Optional. These are example vswitchID values.

5.4.4.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the

trustedCA field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

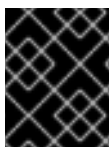


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

5.4.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

5.4.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

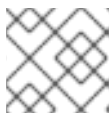
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

5.4.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

5.4.8. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

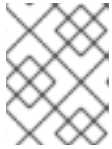
```
$ cat <installation_directory>/auth/kubeadmin-password
```


**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

5.4.9. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

5.4.10. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

5.5. INSTALLING A CLUSTER ON ALIBABA CLOUD WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform 4.12, you can install a cluster on Alibaba Cloud with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

5.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [registered your domain](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud Resource Access Management (RAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain Resource Access Management \(RAM\) credentials](#).

5.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

5.5.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

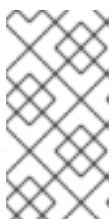
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

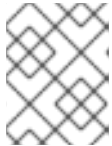
- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

5.5.4. Obtaining the installation program

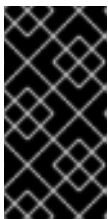
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

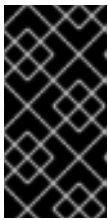
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.5.5. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**

- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.

**NOTE**

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

**IMPORTANT**

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

5.5.5.1. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them

into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

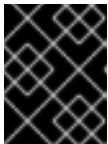
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Enter a descriptive name for your cluster.
 - iii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

5.5.5.2. Generating the required installation manifests

You must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

Procedure

1. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the directory in which the installation program creates files.



NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

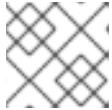
You must have:

- Extracted and prepared the **ccoctl** binary.

Procedure

1. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
<1> `credrequests` is the directory where the list of `CredentialsRequest` objects is stored.
This command creates the directory if it does not exist.
```



NOTE

This command can take a few moments to run.

5.5.5.3. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

5.5.5.3.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.5. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

5.5.5.3.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 5.6. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;">  <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


5.5.5.3.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 5.7. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

5.5.5.4. Sample customized install-config.yaml file for Alibaba Cloud

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: alicloud-dev.devcluster.openshift.com
credentialsMode: Manual
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled

```

```

name: master
platform: {}
replicas: 3
metadata:
  creationTimestamp: null
  name: test-cluster ❶
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes ❷
  serviceNetwork:
  - 172.30.0.0/16
platform:
  alibabacloud:
    defaultMachinePlatform: ❸
      instanceType: ecs.g6.xlarge
      systemDiskCategory: cloud_efficiency
      systemDiskSize: 200
    region: ap-southeast-1 ❹
    resourceGroupID: rg-acfnw6j3hyai ❺
    vpcID: vpc-0xifdjerdibmaqvtjob2b ❻
    vswitchIDs: ❼
      - vsw-0xi8ycgwc8ww5rhviwdq5
      - vsw-0xiy6v3z2tedv009b4pz2
  publish: External
  pullSecret: '{"auths": {"cloud.openshift.com": {"auth": ... }}' ❽
  sshKey: |
    ssh-rsa AAAA... ❾

```

- ❶ Required. The installation program prompts you for a cluster name.
- ❷ The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- ❸ Optional. Specify parameters for machine pools that do not define their own platform configuration.
- ❹ Required. The installation program prompts you for the region to deploy the cluster to.
- ❺ Optional. Specify an existing resource group where the cluster should be installed.
- ❽ Required. The installation program prompts you for the pull secret.
- ❾ Optional. The installation program prompts you for the SSH key value that you use to access the machines in your cluster.
- ❻ ❼ Optional. These are example vswitchID values.

5.5.5.5. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the

trustedCA field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

5.5.6. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

5.5.6.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 5.8. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 5.9. defaultNetwork object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 5.10. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>

Field	Type	Description
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 5.11. ovnKubernetesConfig object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.

Field	Type	Description
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <p> NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 5.12. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 5.13. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 5.14. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

5.5.7. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the `<installation_directory>/manifests/` directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

5.5.8. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.



IMPORTANT

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yaml** in the **<installation_directory>/manifests/** directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yaml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  EOF
```

where:

<installation_directory>

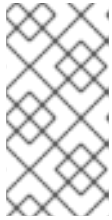
Specifies the directory name that contains the **manifests/** directory for your cluster.

3. Open the **cluster-network-03-config.yaml** file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

Specify a hybrid networking configuration

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
        hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

- 1 Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR cannot overlap with the **clusterNetwork** CIDR.
- 2 Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **4789** port. For more information on this requirement, see the Microsoft documentation on [Pod-to-pod connectivity between hosts is broken](#).



NOTE

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.
5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.

5.5.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

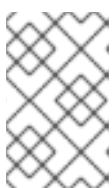
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

5.5.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**. To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

5.5.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

■

```
system:admin
```

5.5.12. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

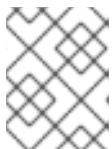


NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https     reencrypt/Redirect    None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

5.5.13. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use](#)

[subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

5.5.14. Next steps

- [Validate an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

5.6. INSTALLING A CLUSTER ON ALIBABA CLOUD INTO AN EXISTING VPC

In OpenShift Container Platform version 4.12, you can install a cluster into an existing Alibaba Virtual Private Cloud (VPC) on Alibaba Cloud Services. The installation program provisions the required infrastructure, which can then be customized. To customize the VPC installation, modify the parameters in the 'install-config.yaml' file before you install the cluster.



NOTE

The scope of the OpenShift Container Platform installation configurations is intentionally narrow. It is designed for simplicity and ensured success. You can complete many more OpenShift Container Platform configuration tasks after an installation completes.



IMPORTANT

Alibaba Cloud on OpenShift Container Platform is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

5.6.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

- You [registered your domain](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud Resource Access Management (RAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain Resource Access Management \(RAM\) credentials](#).

5.6.2. Using a custom VPC

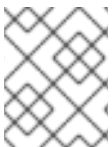
In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Virtual Private Cloud (VPC) in the Alibaba Cloud Platform. By deploying OpenShift Container Platform into an existing Alibaba VPC, you can avoid limit constraints in new accounts and more easily adhere to your organization's operational constraints. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option. You must configure networking using vSwitches.

5.6.2.1. Requirements for using your VPC

The union of the VPC CIDR block and the machine network CIDR must be non-empty. The vSwitches must be within the machine network.

The installation program does not create the following components:

- VPC
- vSwitches
- Route table
- NAT gateway



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

5.6.2.2. VPC validation

To ensure that the vSwitches you provide are suitable, the installation program confirms the following data:

- All the vSwitches that you specify must exist.
- You have provided one or more vSwitches for control plane machines and compute machines.
- The vSwitches' CIDRs belong to the machine CIDR that you specified.

5.6.2.3. Division of permissions

Some individuals can create different resources in your cloud than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components, such as VPCs or vSwitches.

5.6.2.4. Isolation between clusters

If you deploy OpenShift Container Platform into an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed to the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

5.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

5.6.4. Generating a key pair for cluster node SSH access

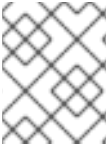
During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

5.6.5. Obtaining the installation program

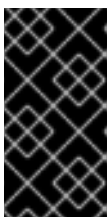
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.6.5.1. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Alibaba Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

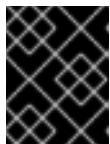
- ii. Select **alibabacloud** as the platform to target.
 - iii. Select the region to deploy the cluster to.
 - iv. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - v. Provide a descriptive name for your cluster.
 - vi. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Installing the cluster into Alibaba Cloud requires that the Cloud Credential Operator (CCO) operate in manual mode. Modify the **install-config.yaml** file to set the **credentialsMode** parameter to **Manual**:

Example **install-config.yaml** configuration file with **credentialsMode** set to **Manual**

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1** Add this line to set the **credentialsMode** to **Manual**.

3. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

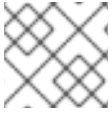


IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

5.6.5.2. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

5.6.5.2.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.15. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

5.6.5.2.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 5.16. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


5.6.5.2.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 5.17. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 593 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 593 1285" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 50px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

5.6.5.2.4. Additional Alibaba Cloud configuration parameters

Additional Alibaba Cloud configuration parameters are described in the following table. The **alibabacloud** parameters are the configuration used when installing on Alibaba Cloud. The **defaultMachinePlatform** parameters are the default configuration used when installing on Alibaba Cloud for machine pools that do not define their own platform configuration.

These parameters apply to both compute machines and control plane machines where specified.



NOTE

If defined, the parameters **compute.platform.alibabacloud** and **controlPlane.platform.alibabacloud** will overwrite **platform.alibabacloud.defaultMachinePlatform** settings for compute machines and control plane machines respectively.

Table 5.18. Optional Alibaba Cloud parameters

Parameter	Description	Values
compute.platform.alibabacLOUD.imageID	The imageID used to create the ECS instance. ImageID must belong to the same region as the cluster.	String.
compute.platform.alibabacLOUD.instanceType	InstanceType defines the ECS instance type. Example: ecs.g6.large	String.
compute.platform.alibabacLOUD.systemDiskCategory	Defines the category of the system disk. Examples: cloud_efficiency,cloud_essd	String.
compute.platform.alibabacLOUD.systemDiskSize	Defines the size of the system disk in gibibytes (GiB).	Integer.
compute.platform.alibabacLOUD.zones	The list of availability zones that can be used. Examples: cn-hangzhou-h, cn-hangzhou-j	String list.
controlPlane.platform.alibabacLOUD.imageID	The imageID used to create the ECS instance. ImageID must belong to the same region as the cluster.	String.
controlPlane.platform.alibabacLOUD.instanceType	InstanceType defines the ECS instance type. Example: ecs.g6.xlarge	String.
controlPlane.platform.alibabacLOUD.systemDiskCategory	Defines the category of the system disk. Examples: cloud_efficiency,cloud_essd	String.
controlPlane.platform.alibabacLOUD.systemDiskSize	Defines the size of the system disk in gibibytes (GiB).	Integer.
controlPlane.platform.alibabacLOUD.zones	The list of availability zones that can be used. Examples: cn-hangzhou-h, cn-hangzhou-j	String list.

Parameter	Description	Values
platform.alibabacloud.region	Required. The Alibaba Cloud region where the cluster will be created.	String.
platform.alibabacloud.resourceGroupID	The ID of an already existing resource group where the cluster will be installed. If empty, the installation program will create a new resource group for the cluster.	String.
platform.alibabacloud.tags	Additional keys and values to apply to all Alibaba Cloud resources created for the cluster.	Object.
platform.alibabacloud.vpcID	The ID of an already existing VPC where the cluster should be installed. If empty, the installation program will create a new VPC for the cluster.	String.
platform.alibabacloud.vswitchIDs	The ID list of already existing VSwitches where cluster resources will be created. The existing VSwitches can only be used when also using existing VPC. If empty, the installation program will create new VSwitches for the cluster.	String list.
platform.alibabacloud.defaultMachinePlatformImageID	For both compute machines and control plane machines, the image ID that should be used to create ECS instance. If set, the image ID should belong to the same region as the cluster.	String.
platform.alibabacloud.defaultMachinePlatformInstanceType	For both compute machines and control plane machines, the ECS instance type used to create the ECS instance. Example: ecs.g6.xlarge	String.

Parameter	Description	Values
platform.alibabacloud.defaultMachinePlatform.systemDiskCategory	For both compute machines and control plane machines, the category of the system disk. Examples: cloud_efficiency , cloud_essd .	String, for example "", cloud_efficiency , cloud_essd .
platform.alibabacloud.defaultMachinePlatform.systemDiskSize	For both compute machines and control plane machines, the size of the system disk in gibibytes (GiB). The minimum is 120 .	Integer.
platform.alibabacloud.defaultMachinePlatform.zones	For both compute machines and control plane machines, the list of availability zones that can be used. Examples: cn-hangzhou-h , cn-hangzhou-j	String list.
platform.alibabacloud.privateZoneID	The ID of an existing private zone into which to add DNS records for the cluster's internal API. An existing private zone can only be used when also using existing VPC. The private zone must be associated with the VPC containing the subnets. Leave the private zone unset to have the installation program create the private zone on your behalf.	String.

5.6.5.3. Sample customized install-config.yaml file for Alibaba Cloud

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: alicloud-dev.devcluster.openshift.com
credentialsMode: Manual
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64

```

```

hyperthreading: Enabled
name: master
platform: {}
replicas: 3
metadata:
  creationTimestamp: null
  name: test-cluster ❶
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes ❷
  serviceNetwork:
    - 172.30.0.0/16
platform:
  alibabacloud:
    defaultMachinePlatform: ❸
    instanceType: ecs.g6.xlarge
    systemDiskCategory: cloud_efficiency
    systemDiskSize: 200
    region: ap-southeast-1 ❹
    resourceGroupID: rg-acfnw6j3hyai ❺
    vpcID: vpc-0xifdjerdibmaqvjob2b ❻
    vswitchIDs: ❼
    - vsw-0xi8ycgwc8wv5rhviwdq5
    - vsw-0xiy6v3z2tedv009b4pz2
  publish: External
  pullSecret: '{"auths": {"cloud.openshift.com": {"auth": ... }}' ❽
  sshKey: |
    ssh-rsa AAAA... ❾

```

- ❶ Required. The installation program prompts you for a cluster name.
- ❷ The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- ❸ Optional. Specify parameters for machine pools that do not define their own platform configuration.
- ❹ Required. The installation program prompts you for the region to deploy the cluster to.
- ❺ Optional. Specify an existing resource group where the cluster should be installed.
- ❽ Required. The installation program prompts you for the pull secret.
- ❾ Optional. The installation program prompts you for the SSH key value that you use to access the machines in your cluster.
- ❻ ❼ Optional. These are example vswitchID values.

5.6.5.4. Generating the required installation manifests

You must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

Procedure

1. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

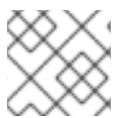
where:

<installation_directory>

Specifies the directory in which the installation program creates files.

5.6.5.5. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.



NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



NOTE

Ensure that the architecture of the **\$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE --file="/usr/bin/ccoctl" -a ~/.pull-secret
```


4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl
```

Verification

- To verify that **ccoctl** is ready to use, display the help file by running the following command:

```
$ ccoctl --help
```

Output of **ccoctl --help**

```
OpenShift credentials provisioning tool
```

```
Usage:
```

```
ccoctl [command]
```

```
Available Commands:
```

```
alibabacloud Manage credentials objects for alibaba cloud
```

```
aws          Manage credentials objects for AWS cloud
```

```
gcp          Manage credentials objects for Google cloud
```

```
help        Help about any command
```

```
ibmcloud    Manage credentials objects for IBM Cloud
```

```
nutanix     Manage credentials objects for Nutanix
```

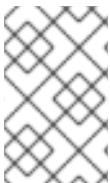
```
Flags:
```

```
-h, --help  help for ccoctl
```

```
Use "ccoctl [command] --help" for more information about a command.
```

5.6.5.6. Creating credentials for OpenShift Container Platform components with the **ccoctl** tool

You can use the OpenShift Container Platform Cloud Credential Operator (CCO) utility to automate the creation of Alibaba Cloud RAM users and policies for each in-cluster component.



NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Created a RAM user with sufficient permission to create the OpenShift Container Platform cluster.
- Added the AccessKeyID (**access_key_id**) and AccessKeySecret (**access_key_secret**) of that RAM user into the `~/alibabacloud/credentials` file on your local computer.

Procedure

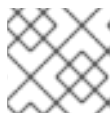
1. Set the **\$RELEASE_IMAGE** variable by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk 'release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
--credentials-requests \
--cloud=alibabacloud \
--to=<path_to_directory_with_list_of_credentials_requests>/credrequests \ 1
$RELEASE_IMAGE
```

- 1** **credrequests** is the directory where the list of **CredentialsRequest** objects is stored. This command creates the directory if it does not exist.



NOTE

This command can take a few moments to run.

3. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on Alibaba Cloud

```
0000_30_machine-api-operator_00_credentials-request.yaml 1
0000_50_cluster-image-registry-operator_01-registry-credentials-request-alibaba.yaml 2
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 3
0000_50_cluster-storage-operator_03_credentials_request_alibaba.yaml 4
```

- 1** The Machine API Operator CR is required.
 - 2** The Image Registry Operator CR is required.
 - 3** The Ingress Operator CR is required.
 - 4** The Storage Operator CR is an optional component and might be disabled in your cluster.
4. Use the **ccoctl** tool to process all **CredentialsRequest** objects in the **credrequests** directory:
 - a. Run the following command to use the tool:

```
$ ccoctl alibabacloud create-ram-users \
--name <name> \
--region=<alibaba_region> \
--credentials-requests-dir=
<path_to_directory_with_list_of_credentials_requests>/credrequests \
--output-dir=<path_to_ccoctl_output_dir>
```

where:

- **<name>** is the name used to tag any cloud resources that are created for tracking.
- **<alibaba_region>** is the Alibaba Cloud region in which cloud resources will be created.
- **<path_to_directory_with_list_of_credentials_requests>/credrequests** is the directory containing the files for the component **CredentialsRequest** objects.
- **<path_to_ccoctl_output_dir>** is the directory where the generated component credentials secrets will be placed.



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

Example output

```
2022/02/11 16:18:26 Created RAM User: user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:27 Ready for creating new ram policy user1-alicloud-openshift-
machine-api-alibabacloud-credentials-policy-policy
2022/02/11 16:18:27 RAM policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has created
2022/02/11 16:18:28 Policy user1-alicloud-openshift-machine-api-alibabacloud-
credentials-policy-policy has attached on user user1-alicloud-openshift-machine-api-
alibabacloud-credentials
2022/02/11 16:18:29 Created access keys for RAM User: user1-alicloud-openshift-
machine-api-alibabacloud-credentials
2022/02/11 16:18:29 Saved credentials configuration to: user1-
alicloud/manifests/openshift-machine-api-alibabacloud-credentials-credentials.yaml
...
```



NOTE

A RAM user can have up to two AccessKeys at the same time. If you run **ccoctl alibabacloud create-ram-users** more than twice, the previous generated manifests secret becomes stale and you must reapply the newly generated secrets.

- Verify that the OpenShift Container Platform secrets are created:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output:

```
openshift-cluster-csi-drivers-alibaba-disk-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-alibabacloud-credentials-credentials.yaml
```

You can verify that the RAM users and policies are created by querying Alibaba Cloud. For more information, refer to Alibaba Cloud documentation on listing RAM users and policies.

- Copy the generated credential files to the target manifests directory:

```
$ cp ./<path_to_ccoctl_output_dir>/manifests/*credentials.yaml
./<path_to_installation_dir>/manifests/
```

where:

<path_to_ccoctl_output_dir>

Specifies the directory created by the **ccoctl alibabacloud create-ram-users** command.

<path_to_installation_dir>

Specifies the directory in which the installation program creates files.

5.6.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

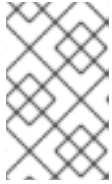
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

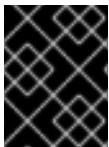
**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

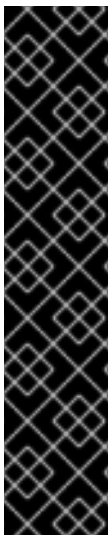
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

5.6.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

5.6.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

5.6.9. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

5.6.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console

5.6.11. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

5.7. UNINSTALLING A CLUSTER ON ALIBABA CLOUD

You can remove a cluster that you deployed to Alibaba Cloud.

5.7.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

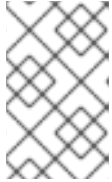
- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$. /openshift-install destroy cluster \  
--dir <installation_directory> --log-level info 1 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

- Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 6. INSTALLING ON AWS

6.1. PREPARING TO INSTALL ON AWS

6.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

6.1.2. Requirements for installing OpenShift Container Platform on AWS

Before installing OpenShift Container Platform on Amazon Web Services (AWS), you must create an AWS account. See [Configuring an AWS account](#) for details about configuring an account, account limits, account permissions, IAM user setup, and supported AWS regions.

If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, see [Manually creating IAM for AWS](#) for other options, including [configuring the Cloud Credential Operator \(CCO\) to use the Amazon Web Services Security Token Service \(AWS STS\)](#).

6.1.3. Choosing a method to install OpenShift Container Platform on AWS

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

6.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on AWS infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- [Installing a cluster quickly on AWS](#) You can install OpenShift Container Platform on AWS infrastructure that is provisioned by the OpenShift Container Platform installation program. You can install a cluster quickly by using the default configuration options.
- [Installing a customized cluster on AWS](#) You can install a customized cluster on AWS infrastructure that the installation program provisions. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).
- [Installing a cluster on AWS with network customizations](#) You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- [Installing a cluster on AWS in a restricted network](#) You can install OpenShift Container Platform on AWS on installer-provisioned infrastructure by using an internal mirror of the

installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components.

- **Installing a cluster on an existing Virtual Private Cloud** You can install OpenShift Container Platform on an existing AWS Virtual Private Cloud (VPC). You can use this installation method if you have constraints set by the guidelines of your company, such as limits when creating new accounts or infrastructure.
- **Installing a private cluster on an existing VPC** You can install a private cluster on an existing AWS VPC. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.
- **Installing a cluster on AWS into a government or secret region** OpenShift Container Platform can be deployed into AWS regions that are specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads in the cloud.

6.1.3.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on AWS infrastructure that you provision, by using one of the following methods:

- **Installing a cluster on AWS infrastructure that you provide** You can install OpenShift Container Platform on AWS infrastructure that you provide. You can use the provided CloudFormation templates to create stacks of AWS resources that represent each of the components required for an OpenShift Container Platform installation.
- **Installing a cluster on AWS in a restricted network with user-provisioned infrastructure** You can install OpenShift Container Platform on AWS infrastructure that you provide by using an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content. While you can install OpenShift Container Platform by using the mirrored content, your cluster still requires internet access to use the AWS APIs.

6.1.4. Next steps

- [Configuring an AWS account](#)

6.2. CONFIGURING AN AWS ACCOUNT

Before you can install OpenShift Container Platform, you must configure an Amazon Web Services (AWS) account.

6.2.1. Configuring Route 53

To install OpenShift Container Platform, the Amazon Web Services (AWS) account you use must have a dedicated public hosted zone in your Route 53 service. This zone must be authoritative for the domain. The Route 53 service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through AWS or another source.

**NOTE**

If you purchase a new domain through AWS, it takes time for the relevant DNS changes to propagate. For more information about purchasing domains through AWS, see [Registering Domain Names Using Amazon Route 53](#) in the AWS documentation.

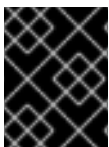
2. If you are using an existing domain and registrar, migrate its DNS to AWS. See [Making Amazon Route 53 the DNS Service for an Existing Domain](#) in the AWS documentation.
3. Create a public hosted zone for your domain or subdomain. See [Creating a Public Hosted Zone](#) in the AWS documentation.
Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
4. Extract the new authoritative name servers from the hosted zone records. See [Getting the Name Servers for a Public Hosted Zone](#) in the AWS documentation.
5. Update the registrar records for the AWS Route 53 name servers that your domain uses. For example, if you registered your domain to a Route 53 service in a different accounts, see the following topic in the AWS documentation: [Adding or Changing Name Servers or Glue Records](#).
6. If you are using a subdomain, add its delegation records to the parent domain. This gives Amazon Route 53 responsibility for the subdomain. Follow the delegation procedure outlined by the DNS provider of the parent domain. See [Creating a subdomain that uses Amazon Route 53 as the DNS service without migrating the parent domain](#) in the AWS documentation for an example high level procedure.

6.2.1.1. Ingress Operator endpoint configuration for AWS Route 53

If you install in either Amazon Web Services (AWS) GovCloud (US) US-West or US-East region, the Ingress Operator uses **us-gov-west-1** region for Route53 and tagging API clients.

The Ingress Operator uses <https://tagging.us-gov-west-1.amazonaws.com> as the tagging API endpoint if a tagging custom endpoint is configured that includes the string 'us-gov-east-1'.

For more information on AWS GovCloud (US) endpoints, see the [Service Endpoints](#) in the AWS documentation about GovCloud (US).

**IMPORTANT**

Private, disconnected installations are not supported for AWS GovCloud when you install in the **us-gov-east-1** region.

Example Route 53 configuration

```
platform:
  aws:
    region: us-gov-west-1
    serviceEndpoints:
      - name: ec2
        url: https://ec2.us-gov-west-1.amazonaws.com
      - name: elasticloadbalancing
        url: https://elasticloadbalancing.us-gov-west-1.amazonaws.com
      - name: route53
```

```
url: https://route53.us-gov.amazonaws.com 1
- name: tagging
url: https://tagging.us-gov-west-1.amazonaws.com 2
```

1 Route 53 defaults to <https://route53.us-gov.amazonaws.com> for both AWS GovCloud (US) regions.


2 Only the US-West region has endpoints for tagging. Omit this parameter if your cluster is in another region.

6.2.2. AWS account limits

The OpenShift Container Platform cluster uses a number of Amazon Web Services (AWS) components, and the default [Service Limits](#) affect your ability to install OpenShift Container Platform clusters. If you use certain cluster configurations, deploy your cluster in certain AWS regions, or run multiple clusters from your account, you might need to request additional resources for your AWS account.

The following table summarizes the AWS components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of clusters available by default	Default AWS limit	Description
Instance Limits	Varies	Varies	<p>By default, each cluster creates the following instances:</p> <ul style="list-style-type: none"> ● One bootstrap machine, which is removed after installation ● Three control plane nodes ● Three worker nodes <p>These instance type counts are within a new account's default limit. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, review your account limits to ensure that your cluster can deploy the machines that you need.</p> <p>In most regions, the worker machines use an m6i.large instance and the bootstrap and control plane machines use m6i.xlarge instances. In some regions, including all regions that do not support these instance types, m5.large and m5.xlarge instances are used instead.</p>

Component	Number of clusters available by default	Default AWS limit	Description
Elastic IPs (EIPs)	0 to 1	5 EIPs per account	<p>To provision the cluster in a highly available configuration, the installation program creates a public and private subnet for each availability zone within a region. Each private subnet requires a NAT Gateway, and each NAT gateway requires a separate elastic IP. Review the AWS region map to determine how many availability zones are in each region. To take advantage of the default high availability, install the cluster in a region with at least three availability zones. To install a cluster in a region with more than five availability zones, you must increase the EIP limit.</p> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>To use the us-east-1 region, you must increase the EIP limit for your account.</p> </div> </div>
Virtual Private Clouds (VPCs)	5	5 VPCs per region	Each cluster creates its own VPC.
Elastic Load Balancing (ELB/NLB)	3	20 per region	By default, each cluster creates internal and external network load balancers for the master API server and a single Classic Load Balancer for the router. Deploying more Kubernetes Service objects with type LoadBalancer will create additional load balancers .
NAT Gateways	5	5 per availability zone	The cluster deploys one NAT gateway in each availability zone.
Elastic Network Interfaces (ENIs)	At least 12	350 per region	<p>The default installation creates 21 ENIs and an ENI for each availability zone in your region. For example, the us-east-1 region contains six availability zones, so a cluster that is deployed in that zone uses 27 ENIs. Review the AWS region map to determine how many availability zones are in each region.</p> <p>Additional ENIs are created for additional machines and ELB load balancers that are created by cluster usage and deployed workloads.</p>

Component	Number of clusters available by default	Default AWS limit	Description
VPC Gateway	20	20 per account	Each cluster creates a single VPC Gateway for S3 access.
S3 buckets	99	100 buckets per account	Because the installation process creates a temporary bucket and the registry component in each cluster creates a bucket, you can create only 99 OpenShift Container Platform clusters per AWS account.
Security Groups	250	2,500 per account	Each cluster creates 10 distinct security groups.

6.2.3. Required AWS permissions for the IAM user



NOTE

Your IAM user must have the permission **tag:GetResources** in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

When you attach the **AdministratorAccess** policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

Example 6.1. Required EC2 permissions for installation

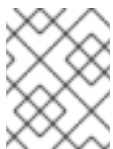
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteSnapshot**
- **ec2>DeleteTags**

- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInstanceTypes**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**

- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

Example 6.2. Required permissions for creating network resources during installation

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



NOTE

If you use an existing VPC, your account does not require these permissions for creating network resources.

Example 6.3. Required Elastic Load Balancing permissions (ELB) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**

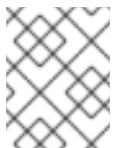
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing>CreateLoadBalancer**
- **elasticloadbalancing>CreateLoadBalancerListeners**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

Example 6.4. Required Elastic Load Balancing permissions (ELBv2) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing>CreateListener**
- **elasticloadbalancing>CreateLoadBalancer**
- **elasticloadbalancing>CreateTargetGroup**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

Example 6.5. Required IAM permissions for installation

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam>DeleteRole**
- **iam>DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**



NOTE

If you have not created a load balancer in your AWS account, the IAM user also requires the **iam:CreateServiceLinkedRole** permission.

Example 6.6. Required Route 53 permissions for installation

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53>DeleteHostedZone**
- **route53:GetChange**

- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

Example 6.7. Required S3 permissions for installation

- **s3:CreateBucket**
- **s3>DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketPolicy**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

Example 6.8. S3 permissions that cluster Operators require

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

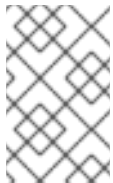
Example 6.9. Required permissions to delete base cluster resources

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeletePlacementGroup**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam>ListAttachedRolePolicies**
- **iam>ListInstanceProfiles**
- **iam>ListRolePolicies**
- **iam>ListUserPolicies**
- **s3:DeleteObject**
- **s3>ListBucketVersions**
- **tag:GetResources**

Example 6.10. Required permissions to delete network resources

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**

- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **ec2:ReplaceRouteTableAssociation**



NOTE

If you use an existing VPC, your account does not require these permissions to delete network resources. Instead, your account only requires the **tag:UntagResources** permission to delete network resources.

Example 6.11. Required permissions to delete a cluster with shared instance roles

- **iam:UntagRole**

Example 6.12. Additional IAM and S3 permissions that are required to create manifests

- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:ListBucket**

- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**



NOTE

If you are managing your cloud provider credentials with mint mode, the IAM user also requires the **iam:CreateAccessKey** and **iam:CreateUser** permissions.

Example 6.13. Optional permissions for instance and quota checks for installation

- **ec2:DescribeInstanceTypeOfferings**
- **servicequotas:ListAWSDefaultServiceQuotas**

6.2.4. Creating an IAM user

Each Amazon Web Services (AWS) account contains a root user account that is based on the email address you used to create the account. This is a highly-privileged account, and it is recommended to use it for only initial account and billing configuration, creating an initial set of users, and securing the account.

Before you install OpenShift Container Platform, create a secondary IAM administrative user. As you complete the [Creating an IAM User in Your AWS Account](#) procedure in the AWS documentation, set the following options:

Procedure

1. Specify the IAM user name and select **Programmatic access**.
2. Attach the **AdministratorAccess** policy to ensure that the account has sufficient permission to create the cluster. This policy provides the cluster with the ability to grant credentials to each OpenShift Container Platform component. The cluster grants the components only the credentials that they require.



NOTE

While it is possible to create a policy that grants the all of the required AWS permissions and attach it to the user, this is not the preferred option. The cluster will not have the ability to grant additional credentials to individual components, so the same credentials are used by all components.

3. Optional: Add metadata to the user by attaching tags.
4. Confirm that the user name that you specified is granted the **AdministratorAccess** policy.
5. Record the access key ID and secret access key values. You must use these values when you configure your local machine to run the installation program.



IMPORTANT

You cannot use a temporary session token that you generated while using a multi-factor authentication device to authenticate to AWS when you deploy a cluster. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials.

Additional resources

- See [Manually creating IAM for AWS](#) for steps to set the Cloud Credential Operator (CCO) to manual mode prior to installation. Use this mode in environments where the cloud identity and access management (IAM) APIs are not reachable, or if you prefer not to store an administrator-level credential secret in the cluster **kube-system** project.

6.2.5. IAM Policies and AWS authentication

By default, the installation program creates instance profiles for the bootstrap, control plane, and compute instances with the necessary permissions for the cluster to operate.

However, you can create your own IAM roles and specify them as part of the installation process. You might need to specify your own roles to deploy the cluster or to manage the cluster after installation. For example:

- Your organization's security policies require that you use a more restrictive set of permissions to install the cluster.
- After the installation, the cluster is configured with an Operator that requires access to additional services.

If you choose to specify your own IAM roles, you can take the following steps:

- Begin with the default policies and adapt as required. For more information, see "Default permissions for IAM instance profiles".
- Use the AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) to create a policy template that is based on the cluster's activity. For more information see, "Using AWS IAM Analyzer to create policy templates".

6.2.5.1. Default permissions for IAM instance profiles

By default, the installation program creates IAM instance profiles for the bootstrap, control plane and worker instances with the necessary permissions for the cluster to operate.

The following lists specify the default permissions for control plane and compute machines:

Example 6.14. Default IAM role permissions for control plane instance profiles

- **ec2:AttachVolume**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**

- **ec2:CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteVolume**
- **ec2:Describe***
- **ec2:DetachVolume**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyVolume**
- **ec2:RevokeSecurityGroupIngress**
- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerPolicy**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing>DeleteListener**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing>DeleteLoadBalancerListeners**
- **elasticloadbalancing>DeleteTargetGroup**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:Describe***
- **elasticloadbalancing:DetachLoadBalancerFromSubnets**
- **elasticloadbalancing:ModifyListener**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**

- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**
- **kms:DescribeKey**

Example 6.15. Default IAM role permissions for compute instance profiles

- **ec2:DescribeInstances**
- **ec2:DescribeRegions**

6.2.5.2. Specifying an existing IAM role

Instead of allowing the installation program to create IAM instance profiles with the default permissions, you can use the **install-config.yaml** file to specify an existing IAM role for control plane and compute instances.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

1. Update **compute.platform.aws.iamRole** with an existing role for the compute machines.

Sample **install-config.yaml** file with an IAM role for compute instances

```
compute:
  - hyperthreading: Enabled
    name: worker
  platform:
    aws:
      iamRole: ExampleRole
```

2. Update **controlPlane.platform.aws.iamRole** with an existing role for the control plane machines.

Sample **install-config.yaml** file with an IAM role for control plane instances

```
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      iamRole: ExampleRole
```

3. Save the file and reference it when installing the OpenShift Container Platform cluster.

**NOTE**

To change or update an IAM account after the cluster has been installed, see [RHOC P 4 AWS cloud-credentials access key is expired](#) (Red Hat Knowledgebase).

Additional resources

- See [Deploying the cluster](#).

6.2.5.3. Using AWS IAM Analyzer to create policy templates

The minimal set of permissions that the control plane and compute instance profiles require depends on how the cluster is configured for its daily operation.

One way to determine which permissions the cluster instances require is to use the AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) to create a policy template:

- A policy template contains the permissions the cluster has used over a specified period of time.
- You can then use the template to create policies with fine-grained permissions.

Procedure

The overall process could be:

1. Ensure that CloudTrail is enabled. CloudTrail records all of the actions and events in your AWS account, including the API calls that are required to create a policy template. For more information, see the AWS documentation for [working with CloudTrail](#).
2. Create an instance profile for control plane instances and an instance profile for compute instances. Be sure to assign each role a permissive policy, such as PowerUserAccess. For more information, see the AWS documentation for [creating instance profile roles](#).
3. Install the cluster in a development environment and configure it as required. Be sure to deploy all of applications the cluster will host in a production environment.
4. Test the cluster thoroughly. Testing the cluster ensures that all of the required API calls are logged.
5. Use the IAM Access Analyzer to create a policy template for each instance profile. For more information, see the AWS documentation for [generating policies based on the CloudTrail logs](#).
6. Create and add a fine-grained policy to each instance profile.
7. Remove the permissive policy from each instance profile.
8. Deploy a production cluster using the existing instance profiles with the new policies.

**NOTE**

You can add [IAM Conditions](#) to your policy to make it more restrictive and compliant with your organization security requirements.

6.2.6. Supported AWS Marketplace regions

Installing an OpenShift Container Platform cluster using an AWS Marketplace image is available to customers who purchase the offer in North America.

While the offer must be purchased in North America, you can deploy the cluster to any of the following supported partitions:

- Public
- GovCloud



NOTE

Deploying a OpenShift Container Platform cluster using an AWS Marketplace image is not supported for the AWS secret regions or China regions.

6.2.7. Supported AWS regions

You can deploy an OpenShift Container Platform cluster to the following regions.



NOTE

Your IAM user must have the permission **tag:GetResources** in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

6.2.7.1. AWS public regions

The following AWS public regions are supported:

- **af-south-1** (Cape Town)
- **ap-east-1** (Hong Kong)
- **ap-northeast-1** (Tokyo)
- **ap-northeast-2** (Seoul)
- **ap-northeast-3** (Osaka)
- **ap-south-1** (Mumbai)
- **ap-south-2** (Hyderabad)
- **ap-southeast-1** (Singapore)
- **ap-southeast-2** (Sydney)
- **ap-southeast-3** (Jakarta)
- **ap-southeast-4** (Melbourne)
- **ca-central-1** (Central)
- **eu-central-1** (Frankfurt)
- **eu-central-2** (Zurich)

- **eu-north-1** (Stockholm)
- **eu-south-1** (Milan)
- **eu-south-2** (Spain)
- **eu-west-1** (Ireland)
- **eu-west-2** (London)
- **eu-west-3** (Paris)
- **me-central-1** (UAE)
- **me-south-1** (Bahrain)
- **sa-east-1** (São Paulo)
- **us-east-1** (N. Virginia)
- **us-east-2** (Ohio)
- **us-west-1** (N. California)
- **us-west-2** (Oregon)

6.2.7.2. AWS GovCloud regions

The following AWS GovCloud regions are supported:

- **us-gov-west-1**
- **us-gov-east-1**

6.2.7.3. AWS SC2S and C2S secret regions

The following AWS secret regions are supported:

- **us-isob-east-1** Secret Commercial Cloud Services (SC2S)
- **us-iso-east-1** Commercial Cloud Services (C2S)

6.2.7.4. AWS China regions

The following AWS China regions are supported:

- **cn-north-1** (Beijing)
- **cn-northwest-1** (Ningxia)

6.2.8. Next steps

- Install an OpenShift Container Platform cluster:
 - [Quickly install a cluster](#) with default options on installer-provisioned infrastructure

- [Install a cluster with cloud customizations on installer-provisioned infrastructure](#)
- [Install a cluster with network customizations on installer-provisioned infrastructure](#)
- [Installing a cluster on user-provisioned infrastructure in AWS by using CloudFormation templates](#)
- [Installing a cluster on AWS with remote workers on AWS Outposts](#)

6.3. MANUALLY CREATING IAM FOR AWS

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

6.3.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the **credentialsMode** parameter in the **install-config.yaml** file.

If you prefer not to store an administrator-level credential secret in the cluster **kube-system** project, you can choose one of the following options when installing OpenShift Container Platform:

- **Use the Amazon Web Services Security Token Service**

You can use the CCO utility (**ccoctl**) to configure the cluster to use the Amazon Web Services Security Token Service (AWS STS). When the CCO utility is used to configure the cluster for STS, it assigns IAM roles that provide short-term, limited-privilege security credentials to components.



NOTE

This credentials strategy is supported for only new OpenShift Container Platform clusters and must be configured during installation. You cannot reconfigure an existing cluster that uses a different credentials strategy to use this feature.

- **Manage cloud credentials manually.**

You can set the **credentialsMode** parameter for the CCO to **Manual** to manage cloud credentials manually. Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

- **Remove the administrator-level credential secret after installing OpenShift Container Platform with mint mode:**

If you are using the CCO with the **credentialsMode** parameter set to **Mint**, you can remove or rotate the administrator-level credential after installing OpenShift Container Platform. Mint mode is the default configuration for the CCO. This option requires the presence of the administrator-level credential during an installation. The administrator-level credential is used during the installation to mint other credentials with some permissions granted. The original credential secret is not stored in the cluster permanently.

**NOTE**

Prior to a non z-stream upgrade, you must reinstate the credential secret with the administrator-level credential. If the credential is not present, the upgrade might be blocked.

Additional resources

- To learn how to use the CCO utility (**ccoctl**) to configure the CCO to use the AWS STS, see [Using manual mode with STS](#).
- To learn how to rotate or remove the administrator-level credential secret after installing OpenShift Container Platform, see [Rotating or removing cloud provider credentials](#).
- For a detailed description of all available CCO credential modes and their supported platforms, see [About the Cloud Credential Operator](#).

6.3.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. Change to the directory that contains the installation program and create the **install-config.yaml** file by running the following command:

```
$ openshift-install create install-config --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

2. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

3. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```


where **<installation_directory>** is the directory in which the installation program creates files.

- From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use by running the following command:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

- Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 \
  --credentials-requests \
  --cloud=aws
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSPProviderSpec
    statementEntries:
      - effect: Allow
        action:
          - iam:GetUser
          - iam:GetUserPolicy
          - iam:ListAccessKeys
        resource: "*"
    ...
```

- Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
```

```
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
      - effect: Allow
        action:
          - s3:CreateBucket
          - s3>DeleteBucket
        resource: "*"
    ...
  secretRef:
    name: <component-secret>
    namespace: <component-namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component-secret>
  namespace: <component-namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```



IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- To find **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation, run the following command:

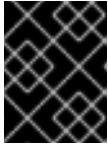
```
$ grep "release.openshift.io/feature-set" *
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```

7. From the directory that contains the installation program, proceed with your cluster creation:

```
$ openshift-install create cluster --dir <installation_directory>
```



IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- [Updating a cluster using the web console](#)
- [Updating a cluster using the CLI](#)

6.3.3. Mint mode

Mint mode is the default Cloud Credential Operator (CCO) credentials mode for OpenShift Container Platform on platforms that support it. In this mode, the CCO uses the provided administrator-level cloud credential to run the cluster. Mint mode is supported for AWS and GCP.

In mint mode, the **admin** credential is stored in the **kube-system** namespace and then used by the CCO to process the **CredentialsRequest** objects in the cluster and create users for each with specific permissions.

The benefits of mint mode include:

- Each cluster component has only the permissions it requires
- Automatic, on-going reconciliation for cloud credentials, including additional credentials or permissions that might be required for upgrades

One drawback is that mint mode requires **admin** credential storage in a cluster **kube-system** secret.

6.3.4. Mint mode with removal or rotation of the administrator-level credential

Currently, this mode is only supported on AWS and GCP.

In this mode, a user installs OpenShift Container Platform with an administrator-level credential just like the normal mint mode. However, this process removes the administrator-level credential secret from the cluster post-installation.

The administrator can have the Cloud Credential Operator make its own request for a read-only credential that allows it to verify if all **CredentialsRequest** objects have their required permissions, thus the administrator-level credential is not required unless something needs to be changed. After the associated credential is removed, it can be deleted or deactivated on the underlying cloud, if desired.



NOTE

Prior to a non z-stream upgrade, you must reinstate the credential secret with the administrator-level credential. If the credential is not present, the upgrade might be blocked.

The administrator-level credential is not stored in the cluster permanently.

Following these steps still requires the administrator-level credential in the cluster for brief periods of time. It also requires manually re-instating the secret with administrator-level credentials for each upgrade.

6.3.5. Next steps

- Install an OpenShift Container Platform cluster:
 - [Installing a cluster quickly on AWS](#) with default options on installer-provisioned infrastructure
 - [Install a cluster with cloud customizations on installer-provisioned infrastructure](#)
 - [Install a cluster with network customizations on installer-provisioned infrastructure](#)
 - [Installing a cluster on user-provisioned infrastructure in AWS by using CloudFormation templates](#)

6.4. INSTALLING A CLUSTER QUICKLY ON AWS

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) that uses the default configuration options.

6.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

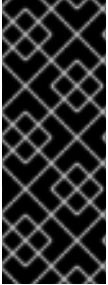
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#). Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

6.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.4.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

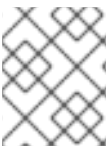
After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

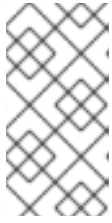
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.4.4. Obtaining the installation program

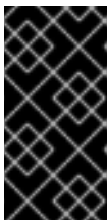
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

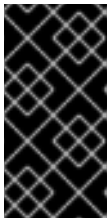
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.4.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Provide values at the prompts:
 - a. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- b. Select **aws** as the platform to target.

- c. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

**NOTE**

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

- d. Select the AWS region to deploy the cluster to.
- e. Select the base domain for the Route 53 service that you configured for your cluster.
- f. Enter a descriptive name for your cluster.
- g. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.

**NOTE**

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
```

```
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Additional resources

- See [Configuration and credential file settings](#) in the AWS documentation for more information about AWS profile and credential configuration.

6.4.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

- Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the appropriate version from the **Version** drop-down list.
- Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
- Unzip the archive with a ZIP program.
- Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

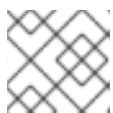
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the appropriate version from the **Version** drop-down list.
- Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

- Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.4.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.4.8. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

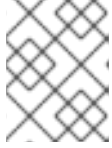
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.4.9. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

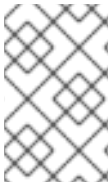
- See [About remote health monitoring](#) for more information about the Telemetry service

6.4.10. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

6.5. INSTALLING A CLUSTER ON AWS WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on infrastructure that the installation program provisions on Amazon Web Services (AWS). To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

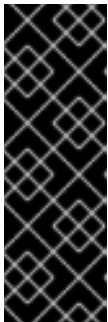


NOTE

The scope of the OpenShift Container Platform installation configurations is intentionally narrow. It is designed for simplicity and ensured success. You can complete many more OpenShift Container Platform configuration tasks after an installation completes.

6.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

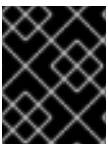
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.5.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

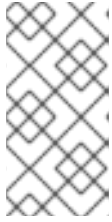
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1** Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.5.4. Obtaining an AWS Marketplace image

If you are deploying an OpenShift Container Platform cluster using an AWS Marketplace image, you must first subscribe through AWS. Subscribing to the offer provides you with the AMI ID that the installation program uses to deploy worker nodes.

Prerequisites

- You have an AWS account to purchase the offer. This account does not have to be the same account that is used to install the cluster.

Procedure

1. Complete the OpenShift Container Platform subscription from the [AWS Marketplace](#).
2. Record the AMI ID for your specific region. As part of the installation process, you must update the **install-config.yaml** file with this value before deploying the cluster.

Sample install-config.yaml file with AWS Marketplace worker nodes

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 1
      type: m5.4xlarge
    replicas: 3
  metadata:
    name: test-cluster
  platform:
    aws:
      region: us-east-2 2
  sshKey: ssh-ed25519 AAAA...
  pullSecret: '{"auths": ...}'
```

- 1** The AMI ID from your AWS Marketplace subscription.
- 2** Your AMI ID is associated with a specific AWS region. When creating the installation configuration file, ensure that you select the same AWS region that you specified when configuring your subscription.

6.5.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.5.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.

- a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
- Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- Select **AWS** as the platform to target.
 - If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.
 - Select the AWS region to deploy the cluster to.
 - Select the base domain for the Route 53 service that you configured for your cluster.
 - Enter a descriptive name for your cluster.
 - Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
- Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 - Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

6.5.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.5.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.1. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.5.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.2. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

6.5.6.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 6.3. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 593 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 593 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	


Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.5.6.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.4. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.plattform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.plattform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.plattform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.plattform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.5.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.5. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.5.6.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.16. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***

- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

6.5.6.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

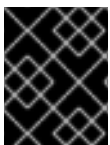
Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.17. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

6.5.6.5. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

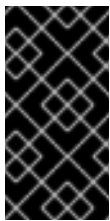
This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
```

```
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
        type: m6i.xlarge
      replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - us-west-2c
      replicas: 3
  metadata:
    name: test-cluster 12
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
        hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 13
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    aws:
      region: us-west-2 14
      propagateUserTags: true 15
      userTags:
        adminContact: jdoe
        costCenter: 7536
      amiID: ami-96c6f8f7 16
      serviceEndpoints: 17
        - name: ec2
          url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
```

```
fips: false 18
sshKey: ssh-ed25519 AAAA... 19
pullSecret: '{"auths": ...}' 20
```

- 1 12 14 20** Required. The installation program prompts you for this value.
- 2** Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.
- 3 8 15** If you do not provide these parameters and values, the installation program provides the default value.
- 4** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 5 9** Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.
- 7 11** Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 13** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16** The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 17** The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 18** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

19

You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

6.5.6.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
```

```

<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2,Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

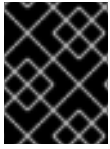
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.5.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

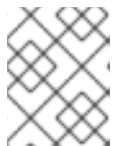
- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

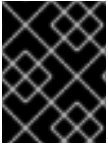
The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.

- Credential information also outputs to `<installation_directory>/openshift_install.log`.

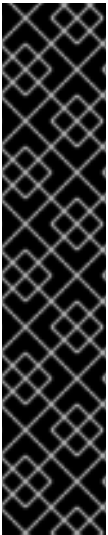


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

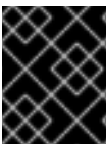


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.5.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.

3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

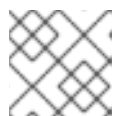
Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.5.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.5.10. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

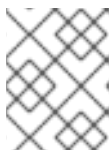


NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.5.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use](#)

[subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.5.12. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

6.6. INSTALLING A CLUSTER ON AWS WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

6.6.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .

6.6.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.6.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

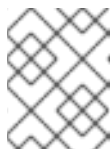
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.6.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.6.5. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

6.6.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.

- a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **AWS** as the platform to target.
- iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.
- iv. Select the AWS region to deploy the cluster to.
- v. Select the base domain for the Route 53 service that you configured for your cluster.
- vi. Enter a descriptive name for your cluster.
- vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

6.6.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.6.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.6. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.6.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.7. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

6.6.6.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 6.8. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: #f0f0f0; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true


Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.6.6.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.9. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.6.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.10. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.6.6.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.18. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***

- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

6.6.6.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.19. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

6.6.6.5. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
```

```

controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
        type: m6i.xlarge
      replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - us-west-2c
      replicas: 3
metadata:
  name: test-cluster 12
networking: 13
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 14
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 15
    propagateUserTags: true 16
    userTags:
      adminContact: jdoe
      costCenter: 7536
    amiID: ami-96c6f8f7 17
    serviceEndpoints: 18
    - name: ec2
      url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com

```

```
fips: false 19
sshKey: ssh-ed25519 AAAA... 20
pullSecret: '{"auths": ...}' 21
```

- 1 12 15 21** Required. The installation program prompts you for this value.
- 2** Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.
- 3 8 13 16** If you do not provide these parameters and values, the installation program provides the default value.
- 4** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 5 9** Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.
- 7 11** Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 14** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 17** The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 18** The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 19** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 20** You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

6.6.6.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
```

```

<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2,Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.6.7. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

6.6.7.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 6.11. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 6.12. defaultNetwork object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 6.13. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 6.14. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 6.15. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 6.16. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 6.17. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

6.6.8. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the `<installation_directory>/manifests/` directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.



NOTE

For more information on using a Network Load Balancer (NLB) on AWS, see [Configuring Ingress cluster traffic on AWS using a Network Load Balancer](#).

6.6.9. Configuring an Ingress Controller Network Load Balancer on a new AWS cluster

You can create an Ingress Controller backed by an AWS Network Load Balancer (NLB) on a new cluster.

Prerequisites

- Create the **install-config.yaml** file and complete any modifications to it.

Procedure

Create an Ingress Controller backed by an AWS NLB on a new cluster.

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a file that is named **cluster-ingress-default-ingresscontroller.yaml** in the **<installation_directory>/manifests/** directory:

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

- 1 For **<installation_directory>**, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

Example output

```
cluster-ingress-default-ingresscontroller.yaml
```

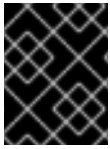
3. Open the **cluster-ingress-default-ingresscontroller.yaml** file in an editor and enter a custom resource (CR) that describes the Operator configuration you want:

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
      providerParameters:
        type: AWS
      aws:
        type: NLB
    type: LoadBalancerService
```

4. Save the **cluster-ingress-default-ingresscontroller.yaml** file and quit the text editor.
5. Optional: Back up the **manifests/cluster-ingress-default-ingresscontroller.yaml** file. The installation program deletes the **manifests/** directory when creating the cluster.

6.6.10. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.



IMPORTANT

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

- Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

where:

<installation_directory>

Specifies the directory name that contains the **manifests/** directory for your cluster.

- Open the **cluster-network-03-config.yml** file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

Specify a hybrid networking configuration

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
```

```

ovnKubernetesConfig:
  hybridOverlayConfig:
    hybridClusterNetwork: 1
    - cidr: 10.132.0.0/14
    hostPrefix: 23
    hybridOverlayVXLANPort: 9898 2

```

- 1** Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR cannot overlap with the **clusterNetwork** CIDR.
- 2** Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **4789** port. For more information on this requirement, see the Microsoft documentation on [Pod-to-pod connectivity between hosts is broken](#).



NOTE

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.
5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.

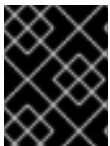


NOTE

For more information on using Linux and Windows nodes in the same cluster, see [Understanding Windows container workloads](#).

6.6.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

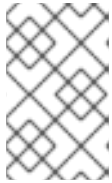
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

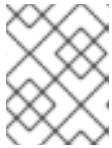
- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.6.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

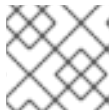
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.6.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.6.14. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

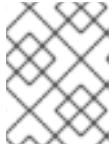
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.6.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.6.16. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

6.7. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) in a restricted network by creating an internal mirror of the installation release content on an existing Amazon Virtual Private Cloud (VPC).

6.7.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [mirrored the images for a disconnected installation](#) to your registry and obtained the `imageContentSources` data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You have an existing VPC in AWS. When installing to a restricted network using installer-provisioned infrastructure, you cannot use the installer-provisioned VPC. You must use a user-provisioned VPC that satisfies one of the following requirements:
 - Contains the mirror registry
 - Has firewall rules or a peering connection to access the mirror registry hosted elsewhere
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#) in the AWS documentation.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

If you are configuring a proxy, be sure to also review this site list.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.7.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

6.7.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

6.7.3. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.7.3.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs

- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.
Public subnets	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.
Internet gateway	<ul style="list-style-type: none"> • AWS::EC2::InternetGateway • AWS::EC2::VPCGatewayAttachment • AWS::EC2::RouteTable • AWS::EC2::Route • AWS::EC2::SubnetRouteTableAssociation • AWS::EC2::NatGateway • AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.

Component	AWS type	Description	
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	You must allow the VPC to access the following ports:	
		Port	Reason
		80	Inbound HTTP traffic
		443	Inbound HTTPS traffic
		22	Inbound SSH traffic
		1024 - 65535	Inbound ephemeral traffic
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.	
		0 - 65535	Outbound ephemeral traffic

6.7.3.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.7.3.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.7.3.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

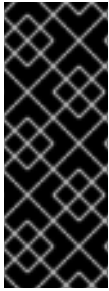
- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.7.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

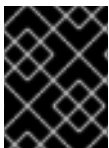
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.7.5. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.7.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **AWS** as the platform to target.
- iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.
- iv. Select the AWS region to deploy the cluster to.
- v. Select the base domain for the Route 53 service that you configured for your cluster.

- vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

  /-----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Define the subnets for the VPC to install the cluster in:

```
subnets:
- subnet-1
- subnet-2
- subnet-3
```

- d. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

3. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

6.7.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.7.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.18. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform , <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.7.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.19. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

6.7.6.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 6.20. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code>
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true


Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.7.6.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.21. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div style="flex-grow: 1;"> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.7.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.22. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.7.6.3. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
hyperthreading: Enabled 5
name: master
platform:
  aws:
    zones:
      - us-west-2a
      - us-west-2b

```



```

rootVolume:
  iops: 4000
  size: 500
  type: io1 6
metadataService:
  authentication: Optional 7
  type: m6i.xlarge
replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - us-west-2c
      replicas: 3
  metadata:
    name: test-cluster 12
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 13
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    aws:
      region: us-west-2 14
      propagateUserTags: true 15
      userTags:
        adminContact: jdoe
        costCenter: 7536
      subnets: 16
      - subnet-1
      - subnet-2
      - subnet-3
      amiID: ami-96c6f8f7 17
      serviceEndpoints: 18
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
      hostedZone: Z3URY6TWQ91KVV 19
  fips: false 20
  sshKey: ssh-ed25519 AAAA... 21
  pullSecret: '{"auths":{"<local_registry>":{"auth": "<credentials>","email": "you@example.com"}}}' 22
  additionalTrustBundle: | 23

```

```
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
```

```
imageContentSources: 24
- mirrors:
- <local_registry>/<local_repository_name>/release
source: quay.io/openshift-release-dev/ocp-release
- mirrors:
- <local_registry>/<local_repository_name>/release
source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

- 1 12 14 Required. The installation program prompts you for this value.
- 2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.
- 3 8 15 If you do not provide these parameters and values, the installation program provides the default value.
- 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 5 9 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10 To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.
- 7 11 Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 13 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16 If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

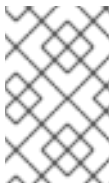
- 17 The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 18 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 19 The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 20 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 21 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 22 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 23 Provide the contents of the certificate file that you used for your mirror registry.
- 24 Provide the **imageContentSources** section from the output of the command to mirror the repository.

6.7.6.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to

hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

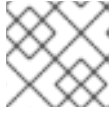
For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2,Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.7.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

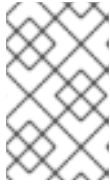
Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1  
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

- Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



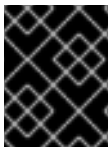
NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

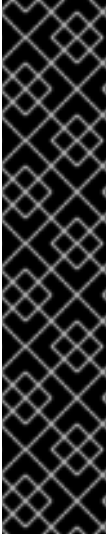


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.7.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

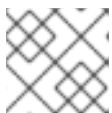
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

6.7.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.7.10. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

6.7.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

6.7.12. Next steps

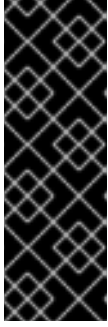
- [Validate an installation](#).
- [Customize your cluster](#).
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#).

6.8. INSTALLING A CLUSTER ON AWS INTO AN EXISTING VPC

In OpenShift Container Platform version 4.12, you can install a cluster into an existing Amazon Virtual Private Cloud (VPC) on Amazon Web Services (AWS). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

6.8.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.8.2. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.8.2.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- Create a public and private subnet for each availability zone that your cluster uses. Each availability zone can contain no more than one public and one private subnet. For an example of this type of configuration, see [VPC with public and private subnets \(NAT\)](#) in the AWS documentation.
Record each subnet ID. Completing the installation requires that you enter these values in the **platform** section of the **install-config.yaml** file. See [Finding a subnet ID](#) in the AWS documentation.
- The VPC's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines. The subnet CIDR blocks must belong to the machine CIDR that you specify.
- The VPC must have a public internet gateway attached to it. For each availability zone:
 - The public subnet requires a route to the internet gateway.
 - The public subnet requires a NAT gateway with an EIP address.
 - The private subnet requires a route to the NAT gateway in public subnet.
- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**

- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.
Public subnets	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.

Component	AWS type	Description												
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	<p>You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.</p>												
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	<p>You must allow the VPC to access the following ports:</p> <table border="1"> <thead> <tr> <th>Port</th> <th>Reason</th> </tr> </thead> <tbody> <tr> <td>80</td> <td>Inbound HTTP traffic</td> </tr> <tr> <td>443</td> <td>Inbound HTTPS traffic</td> </tr> <tr> <td>22</td> <td>Inbound SSH traffic</td> </tr> <tr> <td>1024 - 65535</td> <td>Inbound ephemeral traffic</td> </tr> <tr> <td>0 - 65535</td> <td>Outbound ephemeral traffic</td> </tr> </tbody> </table>	Port	Reason	80	Inbound HTTP traffic	443	Inbound HTTPS traffic	22	Inbound SSH traffic	1024 - 65535	Inbound ephemeral traffic	0 - 65535	Outbound ephemeral traffic
Port	Reason													
80	Inbound HTTP traffic													
443	Inbound HTTPS traffic													
22	Inbound SSH traffic													
1024 - 65535	Inbound ephemeral traffic													
0 - 65535	Outbound ephemeral traffic													
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	<p>Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.</p>												

6.8.2.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.

- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.8.2.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.8.2.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

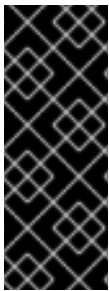
6.8.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

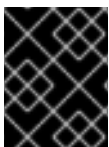
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.8.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

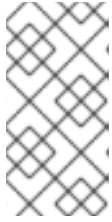
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.8.5. Obtaining the installation program

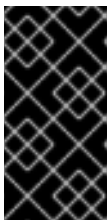
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.8.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **AWS** as the platform to target.
 - iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.
 - iv. Select the AWS region to deploy the cluster to.
 - v. Select the base domain for the Route 53 service that you configured for your cluster.
 - vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

6.8.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.8.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.23. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.8.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.24. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

6.8.6.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 6.25. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1442 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true


Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.8.6.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.26. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.8.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.27. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.8.6.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

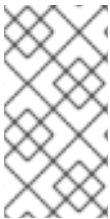
Example 6.20. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***

- m4.*
- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

6.8.6.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.21. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

6.8.6.5. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
```

```
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
        type: m6i.xlarge
      replicas: 3
compute: 8
  - hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - us-west-2c
      replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
```

```

amiID: ami-96c6f8f7 17
serviceEndpoints: 18
  - name: ec2
    url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
hostedZone: Z3URY6TWQ91KVV 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
pullSecret: '{"auths": ...}' 22

```

- 1 12 14 22** Required. The installation program prompts you for this value.
- 2** Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.
- 3 8 15** If you do not provide these parameters and values, the installation program provides the default value.
- 4** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 5 9** Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iosps** to **2000**.
- 7 11** Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.

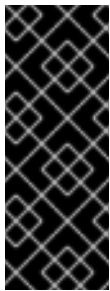


NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 13** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16** If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

- 17 The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 18 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 19 The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 20 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 21 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

6.8.6.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

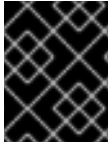
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.8.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$. /openshift-install create cluster --dir <installation_directory> \ 1  
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

- Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



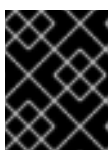
NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.8.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.8.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.8.10. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

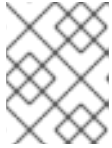
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.8.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.8.12. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

6.9. INSTALLING A PRIVATE CLUSTER ON AWS

In OpenShift Container Platform version 4.12, you can install a private cluster into an existing VPC on Amazon Web Services (AWS). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

6.9.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the `kube-system` namespace, you can [manually create and maintain IAM credentials](#).

6.9.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.
 - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

6.9.2.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for access from only the private network.

The cluster still requires access to internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

6.9.2.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from the internet on 6443 (Kubernetes API port).
- If you use a public Service type load balancer, you must tag a public subnet in each availability zone with **kubernetes.io/cluster/<cluster-infra-id>: shared** so that AWS can use them to create public load balancers.

6.9.3. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.9.3.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.

If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.

Component	AWS type	Description												
Public subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.												
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.												
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkACL ● AWS::EC2::NetworkACLEntry 	<p>You must allow the VPC to access the following ports:</p> <table border="1"> <thead> <tr> <th>Port</th> <th>Reason</th> </tr> </thead> <tbody> <tr> <td>80</td> <td>Inbound HTTP traffic</td> </tr> <tr> <td>443</td> <td>Inbound HTTPS traffic</td> </tr> <tr> <td>22</td> <td>Inbound SSH traffic</td> </tr> <tr> <td>1024 - 65535</td> <td>Inbound ephemeral traffic</td> </tr> <tr> <td>0 - 65535</td> <td>Outbound ephemeral traffic</td> </tr> </tbody> </table>	Port	Reason	80	Inbound HTTP traffic	443	Inbound HTTPS traffic	22	Inbound SSH traffic	1024 - 65535	Inbound ephemeral traffic	0 - 65535	Outbound ephemeral traffic
Port	Reason													
80	Inbound HTTP traffic													
443	Inbound HTTPS traffic													
22	Inbound SSH traffic													
1024 - 65535	Inbound ephemeral traffic													
0 - 65535	Outbound ephemeral traffic													
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.												

6.9.3.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.9.3.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.9.3.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.9.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.9.5. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.9.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

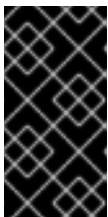
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.9.7. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

6.9.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.9.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.28. Required parameters

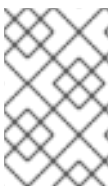
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.9.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.29. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


6.9.7.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 6.30. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="488 192 592 913"></p> <p data-bbox="671 109 932 913">IMPORTANT To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="488 958 592 1155"></p> <p data-bbox="671 965 932 1155">NOTE If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.


Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.9.7.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.31. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
compute.aws.region	The AWS region that the installation program creates compute resources in.	Any valid AWS region , such as us-east-1 . You can use the AWS CLI to access the regions available based on your selected instance type. For example: <pre>aws ec2 describe-instance-type-offerings -- filters Name=instance- type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.

Parameter	Description	Values
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .

Parameter	Description	Values
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	<p>Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.</p> </div> </div>
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .

Parameter	Description	Values
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.9.7.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.32. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.9.7.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.22. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.9.7.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.23. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

6.9.7.5. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
        type: m6i.xlarge
      replicas: 3
compute: 8
  - hyperthreading: Enabled 9
    name: worker
    platform:
      aws:
        rootVolume:
          iops: 2000

```

```

    size: 500
    type: io1 10
  metadataService:
    authentication: Optional 11
    type: c5.4xlarge
    zones:
      - us-west-2c
  replicas: 3
  metadata:
    name: test-cluster 12
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
        hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 13
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    aws:
      region: us-west-2 14
      propagateUserTags: true 15
      userTags:
        adminContact: jdoe
        costCenter: 7536
      subnets: 16
      - subnet-1
      - subnet-2
      - subnet-3
      amiID: ami-96c6f8f7 17
      serviceEndpoints: 18
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
      hostedZone: Z3URY6TWQ91KVV 19
    fips: false 20
    sshKey: ssh-ed25519 AAAA... 21
    publish: Internal 22
    pullSecret: '{"auths": ...}' 23

```

1 12 14 23 Required. The installation program prompts you for this value.

2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.

3 8 15 If you do not provide these parameters and values, the installation program provides the default value.

4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

- 5 9** Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.

- 7 11** Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

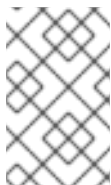
- 13** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16** If you provide your own VPC, specify subnets for each availability zone that your cluster uses.
- 17** The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 18** The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 19** The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 20** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 21** You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 22** How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

6.9.7.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

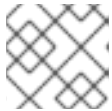
For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

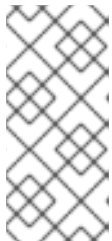
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2,Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.9.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

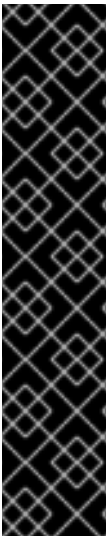


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.9.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**. To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.9.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.9.11. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.9.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.9.13. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials.](#)

6.10. INSTALLING A CLUSTER ON AWS INTO A GOVERNMENT REGION

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) into a government region. To configure the region, modify parameters in the **install-config.yaml** file before you install the cluster.

6.10.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users.](#)
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .

6.10.2. AWS government regions

OpenShift Container Platform supports deploying a cluster to an [AWS GovCloud \(US\)](#) region.

The following AWS GovCloud partitions are supported:

- **us-gov-east-1**
- **us-gov-west-1**

6.10.3. Installation requirements

Before you can install the cluster, you must:

- Provide an existing private AWS VPC and subnets to host the cluster. Public zones are not supported in Route 53 in AWS GovCloud. As a result, clusters must be private when you deploy to an AWS government region.
- Manually create the installation configuration file (**install-config.yaml**).

6.10.4. Private clusters

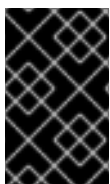
You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.



NOTE

Public zones are not supported in Route 53 in an AWS GovCloud Region. Therefore, clusters must be private if they are deployed to an AWS GovCloud Region.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.
 - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

6.10.4.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for access from only the private network.

The cluster still requires access to internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

6.10.4.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from the internet on 6443 (Kubernetes API port).
- If you use a public Service type load balancer, you must tag a public subnet in each availability zone with **kubernetes.io/cluster/<cluster-infra-id>: shared** so that AWS can use them to create public load balancers.

6.10.5. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.10.5.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options

- VPC endpoints

**NOTE**

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.
Public subnets	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.
Internet gateway	<ul style="list-style-type: none"> • AWS::EC2::InternetGateway • AWS::EC2::VPCGatewayAttachment • AWS::EC2::RouteTable • AWS::EC2::Route • AWS::EC2::SubnetRouteTableAssociation • AWS::EC2::NatGateway • AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.

Component	AWS type	Description	
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	You must allow the VPC to access the following ports:	
		Port	Reason
		80	Inbound HTTP traffic
		443	Inbound HTTPS traffic
		22	Inbound SSH traffic
		1024 - 65535	Inbound ephemeral traffic
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.	
		Port	Reason
		80	Inbound HTTP traffic
		443	Inbound HTTPS traffic
		22	Inbound SSH traffic
		1024 - 65535	Inbound ephemeral traffic
0 - 65535	Outbound ephemeral traffic		

6.10.5.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.10.5.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.10.5.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.10.6. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.10.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```


3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.10.8. Obtaining an AWS Marketplace image

If you are deploying an OpenShift Container Platform cluster using an AWS Marketplace image, you must first subscribe through AWS. Subscribing to the offer provides you with the AMI ID that the installation program uses to deploy worker nodes.

Prerequisites

- You have an AWS account to purchase the offer. This account does not have to be the same account that is used to install the cluster.

Procedure

1. Complete the OpenShift Container Platform subscription from the [AWS Marketplace](#).
2. Record the AMI ID for your specific region. As part of the installation process, you must update the **install-config.yaml** file with this value before deploying the cluster.

Sample install-config.yaml file with AWS Marketplace worker nodes

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 1
      type: m5.4xlarge
    replicas: 3
metadata:
  name: test-cluster
platform:
  aws:
    region: us-east-2 2
sshKey: ssh-ed25519 AAAA...
pullSecret: '{"auths": ...}'
```

- 1** The AMI ID from your AWS Marketplace subscription.
- 2** Your AMI ID is associated with a specific AWS region. When creating the installation configuration file, ensure that you select the same AWS region that you specified when configuring your subscription.

6.10.9. Obtaining the installation program

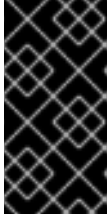
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

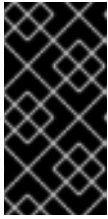
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.10.10. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

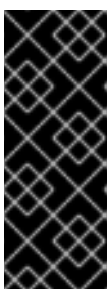
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

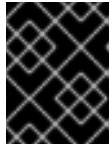
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

6.10.10.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.10.10.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.33. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.10.10.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.34. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


6.10.10.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 6.35. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="border: 1px solid black; width: 20px; height: 20px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="border: 1px solid black; width: 20px; height: 20px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 20px;"> <div style="width: 20px; height: 50px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.


Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

6.10.10.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.36. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set <i>AWS region</i> . See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.10.10.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.37. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.10.10.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.24. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***

- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

6.10.10.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

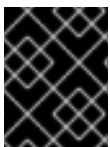
Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.25. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

6.10.10.5. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
```

```
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
        - us-gov-west-1a
        - us-gov-west-1b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
        type: m6i.xlarge
      replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - us-gov-west-1c
      replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-gov-west-1 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
  amiID: ami-96c6f8f7 17
```

```

serviceEndpoints: 18
  - name: ec2
    url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
  hostedZone: Z3URY6TWQ91KVV 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
publish: Internal 22
pullSecret: '{"auths": ...}' 23

```

1 12 14 23 Required.

2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.

3 8 15 If you do not provide these parameters and values, the installation program provides the default value.

4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

5 9 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

6 10 To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iopts** to **2000**.

7 11 Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



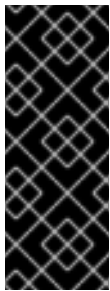
NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

13 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

16 If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

- 17 The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 18 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 19 The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 20 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 21 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 22 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

6.10.10.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

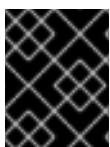
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.10.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$. /openshift-install create cluster --dir <installation_directory> \ 1  
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

- Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

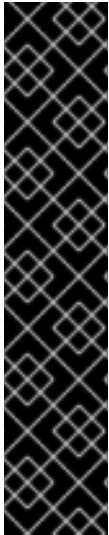


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.10.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.10.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.10.14. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

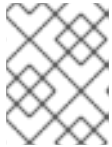
```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.10.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.10.16. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

6.11. INSTALLING A CLUSTER ON AWS INTO A SECRET OR TOP SECRET REGION

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) into the following secret regions:

- Secret Commercial Cloud Services (SC2S)
- Commercial Cloud Services (C2S)

To configure a cluster in either region, you change parameters in the **install config.yaml** file before you install the cluster.

6.11.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multifactor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.11.2. AWS secret regions

The following AWS secret partitions are supported:

- **us-isob-east-1** (SC2S)
- **us-iso-east-1** (C2S)



NOTE

The maximum supported MTU in an AWS SC2S and C2S Regions is not the same as AWS commercial. For more information about configuring MTU during installation, see the *Cluster Network Operator configuration object* section in *Installing a cluster on AWS with network customizations*

6.11.3. Installation requirements

Red Hat does not publish a Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image for the AWS Secret and Top Secret Regions.

Before you can install the cluster, you must:

- Upload a custom RHCOS AMI.
- Manually create the installation configuration file (**install-config.yaml**).
- Specify the AWS region, and the accompanying custom AMI, in the installation configuration file.

You cannot use the OpenShift Container Platform installation program to create the installation configuration file. The installer does not list an AWS region without native support for an RHCOS AMI.



IMPORTANT

You must also define a custom CA certificate in the **additionalTrustBundle** field of the **install-config.yaml** file because the AWS API requires a custom CA trust bundle. To allow the installation program to access the AWS API, the CA certificates must also be defined on the machine that runs the installation program. You must add the CA bundle to the trust store on the machine, use the **AWS_CA_BUNDLE** environment variable, or define the CA bundle in the **ca_bundle** field of the AWS config file.

6.11.4. Private clusters

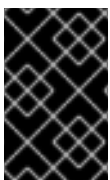
You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.



NOTE

Public zones are not supported in Route 53 in an AWS Top Secret Region. Therefore, clusters must be private if they are deployed to an AWS Top Secret Region.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:

- The API services for the cloud to which you provision.
- The hosts on the network that you provision.
- The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

6.11.4.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for access from only the private network.

The cluster still requires access to internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

6.11.4.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from the internet on 6443 (Kubernetes API port).
- If you use a public Service type load balancer, you must tag a public subnet in each availability zone with **kubernetes.io/cluster/<cluster-infra-id>: shared** so that AWS can use them to create public load balancers.

6.11.5. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.11.5.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

A cluster in an SC2S or C2S Region is unable to reach the public IP addresses for the EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

SC2S

- **elasticloadbalancing.<aws_region>.sc2s.sgov.gov**
- **ec2.<aws_region>.sc2s.sgov.gov**
- **s3.<aws_region>.sc2s.sgov.gov**

C2S

- **elasticloadbalancing.<aws_region>.c2s.ic.gov**
- **ec2.<aws_region>.c2s.ic.gov**
- **s3.<aws_region>.c2s.ic.gov**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

SC2S

- **elasticloadbalancing.<aws_region>.sc2s.sgov.gov**
- **ec2.<aws_region>.sc2s.sgov.gov**
- **s3.<aws_region>.sc2s.sgov.gov**

C2S

- **elasticloadbalancing.<aws_region>.c2s.ic.gov**
- **ec2.<aws_region>.c2s.ic.gov**
- **s3.<aws_region>.c2s.ic.gov**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description												
VPC	<ul style="list-style-type: none"> ● AWS::EC2::VPC ● AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.												
Public subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.												
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.												
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkACL ● AWS::EC2::NetworkACLEntry 	<p>You must allow the VPC to access the following ports:</p> <table border="1"> <thead> <tr> <th>Port</th> <th>Reason</th> </tr> </thead> <tbody> <tr> <td>80</td> <td>Inbound HTTP traffic</td> </tr> <tr> <td>443</td> <td>Inbound HTTPS traffic</td> </tr> <tr> <td>22</td> <td>Inbound SSH traffic</td> </tr> <tr> <td>1024 - 65535</td> <td>Inbound ephemeral traffic</td> </tr> <tr> <td>0 - 65535</td> <td>Outbound ephemeral traffic</td> </tr> </tbody> </table>	Port	Reason	80	Inbound HTTP traffic	443	Inbound HTTPS traffic	22	Inbound SSH traffic	1024 - 65535	Inbound ephemeral traffic	0 - 65535	Outbound ephemeral traffic
Port	Reason													
80	Inbound HTTP traffic													
443	Inbound HTTPS traffic													
22	Inbound SSH traffic													
1024 - 65535	Inbound ephemeral traffic													
0 - 65535	Outbound ephemeral traffic													

Component	AWS type	Description
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

6.11.5.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.11.5.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.11.5.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.11.6. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.11.7. Uploading a custom RHCOS AMI in AWS

If you are deploying to a custom Amazon Web Services (AWS) region, you must upload a custom Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) that belongs to that region.

Prerequisites

- You configured an AWS account.
- You created an Amazon S3 bucket with the required IAM [service role](#).
- You uploaded your RHCOS VMDK file to Amazon S3. The RHCOS VMDK file must be the highest version that is less than or equal to the OpenShift Container Platform version you are installing.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer](#).

Procedure

1. Export your AWS profile as an environment variable:

```
$ export AWS_PROFILE=<aws_profile> ❶
```

- Export the region to associate with your custom AMI as an environment variable:

```
$ export AWS_DEFAULT_REGION=<aws_region> ❶
```

- Export the version of RHCOS you uploaded to Amazon S3 as an environment variable:

```
$ export RHCOS_VERSION=<version> ❶
```

❶ ❶ ❶ The RHCOS VMDK version, like **4.12.0**.

- Export the Amazon S3 bucket name as an environment variable:

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

- Create the **containers.json** file and define your RHCOS VMDK file:

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

- Import the RHCOS disk as an Amazon EBS snapshot:

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
  --description "<description>" ❶
  --disk-container "file://<file_path>/containers.json" ❷
```

❶ The description of your RHCOS disk being imported, like **rhcos-\${RHCOS_VERSION}-x86_64-aws.x86_64**.

❷ The file path to the JSON file describing your RHCOS disk. The JSON file should contain your Amazon S3 bucket name and key.

- Check the status of the image import:

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

Example output

```
{
  "ImportSnapshotTasks": [
    {
      "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
```

```

    "ImportTaskId": "import-snap-fh6i8uil",
    "SnapshotTaskDetail": {
      "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
      "DiskImageSize": 819056640.0,
      "Format": "VMDK",
      "SnapshotId": "snap-06331325870076318",
      "Status": "completed",
      "UserBucket": {
        "S3Bucket": "external-images",
        "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
      }
    }
  }
]
}

```

Copy the **SnapshotId** to register the image.

8. Create a custom RHCOS AMI from the RHCOS snapshot:

```

$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4

```

- 1** The RHCOS VMDK architecture type, like **x86_64**, **aarch64**, **s390x**, or **ppc64le**.
- 2** The **Description** from the imported snapshot.
- 3** The name of the RHCOS AMI.
- 4** The **SnapshotID** from the imported snapshot.

To learn more about these APIs, see the AWS documentation for [importing snapshots](#) and [creating EBS-backed AMIs](#).

6.11.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

-


```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.11.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.11.10. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have uploaded a custom RHCOS AMI.
- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

6.11.10.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.11.10.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.38. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

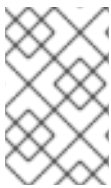
Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.11.10.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.39. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


6.11.10.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 6.40. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="488 192 592 913"></p> <p data-bbox="671 107 935 913">IMPORTANT To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="488 958 592 1155"></p> <p data-bbox="671 965 935 1155">NOTE If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.


Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.11.10.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.41. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
compute.aws.region	The AWS region that the installation program creates compute resources in.	Any valid AWS region , such as us-east-1 . You can use the AWS CLI to access the regions available based on your selected instance type. For example: <pre>aws ec2 describe-instance-type-offerings -- filters Name=instance- type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.platform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.platform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.

Parameter	Description	Values
controlPlane.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.platform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .

Parameter	Description	Values
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	<p>Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.</p> </div> </div>
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .

Parameter	Description	Values
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.11.10.2. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.26. Machine types based on 64-bit x86 architecture for secret regions

- c4.*
- c5.*
- i3.*
- m4.*
- m5.*
- r4.*
- r5.*
- t3.*

6.11.10.3. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```

apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
      - us-iso-east-1a
      - us-iso-east-1b
    rootVolume:
      iops: 4000
      size: 500
      type: io1 6
    metadataService:
      authentication: Optional 7
      type: m6i.xlarge
    replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
      - us-iso-east-1a
      - us-iso-east-1b
    replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
networkType: OVNKubernetes 13
serviceNetwork:
- 172.30.0.0/16
platform:
  aws:
    region: us-iso-east-1 14

```

```

propagateUserTags: true 15
userTags:
  adminContact: jdoe
  costCenter: 7536
subnets: 16
- subnet-1
- subnet-2
- subnet-3
amiID: ami-96c6f8f7 17 18
serviceEndpoints: 19
- name: ec2
  url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
hostedZone: Z3URY6TWQ91KVV 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22
publish: Internal 23
pullSecret: '{"auths": ...}' 24
additionalTrustBundle: | 25
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----

```

1 12 14 17 24 Required.

2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.

3 8 15 If you do not provide these parameters and values, the installation program provides the default value.

4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

5 9 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

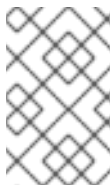


IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

6 10 To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.

7 11 Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2,

**NOTE**

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 13 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16 If you provide your own VPC, specify subnets for each availability zone that your cluster uses.
- 18 The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.
- 19 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 20 The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 21 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 22 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 23 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.
- 25 The custom CA certificate. This is required when deploying to the SC2S or C2S Regions because the AWS API requires a custom CA trust bundle.

6.11.10.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then

creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.11.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

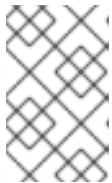
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

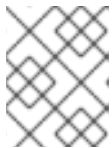
- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



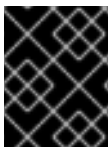
NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.11.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```


-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.11.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.11.14. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

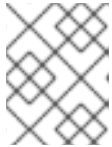
```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- [Accessing the web console](#)

6.11.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

6.11.16. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

6.12. INSTALLING A CLUSTER ON AWS CHINA

In OpenShift Container Platform version 4.12, you can install a cluster to the following Amazon Web Services (AWS) China regions:

- **cn-north-1** (Beijing)
- **cn-northwest-1** (Ningxia)

6.12.1. Prerequisites

- You have an Internet Content Provider (ICP) license.
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

6.12.2. Installation requirements

Red Hat does not publish a Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) for the AWS China regions.

Before you can install the cluster, you must:

- Upload a custom RHCOS AMI.
- Manually create the installation configuration file (**install-config.yaml**).
- Specify the AWS region, and the accompanying custom AMI, in the installation configuration file.

You cannot use the OpenShift Container Platform installation program to create the installation configuration file. The installer does not list an AWS region without native support for an RHCOS AMI.

6.12.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



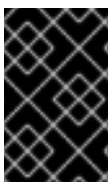
IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.12.4. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.
 - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network.

**NOTE**

AWS China does not support a VPN connection between the VPC and your network. For more information about the Amazon VPC service in the Beijing and Ningxia regions, see [Amazon Virtual Private Cloud](#) in the AWS China documentation.

6.12.4.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for access from only the private network.

The cluster still requires access to internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

6.12.4.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from the internet on 6443 (Kubernetes API port).
- If you use a public Service type load balancer, you must tag a public subnet in each availability zone with **kubernetes.io/cluster/<cluster-infra-id>: shared** so that AWS can use them to create public load balancers.

6.12.5. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.12.5.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the **kubernetes.io/cluster/.*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/.*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com.cn**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com.cn**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.
Public subnets	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.

Component	AWS type	Description												
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	<p>You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.</p>												
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	<p>You must allow the VPC to access the following ports:</p> <table border="1"> <thead> <tr> <th>Port</th> <th>Reason</th> </tr> </thead> <tbody> <tr> <td>80</td> <td>Inbound HTTP traffic</td> </tr> <tr> <td>443</td> <td>Inbound HTTPS traffic</td> </tr> <tr> <td>22</td> <td>Inbound SSH traffic</td> </tr> <tr> <td>1024 - 65535</td> <td>Inbound ephemeral traffic</td> </tr> <tr> <td>0 - 65535</td> <td>Outbound ephemeral traffic</td> </tr> </tbody> </table>	Port	Reason	80	Inbound HTTP traffic	443	Inbound HTTPS traffic	22	Inbound SSH traffic	1024 - 65535	Inbound ephemeral traffic	0 - 65535	Outbound ephemeral traffic
Port	Reason													
80	Inbound HTTP traffic													
443	Inbound HTTPS traffic													
22	Inbound SSH traffic													
1024 - 65535	Inbound ephemeral traffic													
0 - 65535	Outbound ephemeral traffic													
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	<p>Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.</p>												

6.12.5.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.12.5.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.12.5.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.12.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

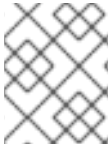
After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.12.7. Uploading a custom RHCOS AMI in AWS

If you are deploying to a custom Amazon Web Services (AWS) region, you must upload a custom Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) that belongs to that region.

Prerequisites

- You configured an AWS account.
- You created an Amazon S3 bucket with the required IAM [service role](#).
- You uploaded your RHCOS VMDK file to Amazon S3. The RHCOS VMDK file must be the highest version that is less than or equal to the OpenShift Container Platform version you are installing.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer](#).

Procedure

1. Export your AWS profile as an environment variable:

```
$ export AWS_PROFILE=<aws_profile> 1
```

- 1 The AWS profile name that holds your AWS credentials, like **beijingadmin**.

2. Export the region to associate with your custom AMI as an environment variable:

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

- 1 The AWS region, like **cn-north-1**.

3. Export the version of RHCOS you uploaded to Amazon S3 as an environment variable:

```
$ export RHCOS_VERSION=<version> 1
```

- 1 The RHCOS VMDK version, like **4.12.0**.

4. Export the Amazon S3 bucket name as an environment variable:

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. Create the **containers.json** file and define your RHCOS VMDK file:

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos- $\{RHCOS\_VERSION\}$ -x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": " $\{VMIMPORT\_BUCKET\_NAME\}$ ",
    "S3Key": "rhcos- $\{RHCOS\_VERSION\}$ -x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. Import the RHCOS disk as an Amazon EBS snapshot:

```
$ aws ec2 import-snapshot --region  $\{AWS\_DEFAULT\_REGION\}$  \
  --description "<description>" 1 \
  --disk-container "file://<file_path>/containers.json" 2
```

- 1 The description of your RHCOS disk being imported, like **rhcos- $\{RHCOS_VERSION\}$ -x86_64-aws.x86_64**.

- 2 The file path to the JSON file describing your RHCOS disk. The JSON file should contain your Amazon S3 bucket name and key.

7. Check the status of the image import:

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region  $\{AWS\_DEFAULT\_REGION\}$ 
```

Example output

```
{
  "ImportSnapshotTasks": [
    {
      "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
      "ImportTaskId": "import-snap-fh6i8uil",
      "SnapshotTaskDetail": {
        "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
        "DiskImageSize": 819056640.0,
        "Format": "VMDK",
        "SnapshotId": "snap-06331325870076318",
        "Status": "completed",
        "UserBucket": {
          "S3Bucket": "external-images",
          "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
        }
      }
    }
  ]
}
```

Copy the **SnapshotId** to register the image.

8. Create a custom RHCOS AMI from the RHCOS snapshot:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

- 1** The RHCOS VMDK architecture type, like **x86_64**, **aarch64**, **s390x**, or **ppc64le**.
- 2** The **Description** from the imported snapshot.
- 3** The name of the RHCOS AMI.
- 4** The **SnapshotID** from the imported snapshot.

To learn more about these APIs, see the AWS documentation for [importing snapshots](#) and [creating EBS-backed AMIs](#).

6.12.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.12.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

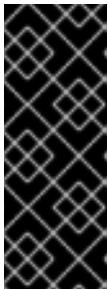
Prerequisites

- You have uploaded a custom RHCOS AMI.
- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

6.12.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.12.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.42. Required parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.12.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.43. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


6.12.9.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 6.44. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div style="margin-left: 10px;"> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

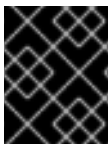
Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 30px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.12.9.2. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```

apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
name: master

```

```
platform:
  aws:
    zones:
      - cn-north-1a
      - cn-north-1b
    rootVolume:
      iops: 4000
      size: 500
      type: io1 6
    metadataService:
      authentication: Optional 7
      type: m6i.xlarge
    replicas: 3
  compute: 8
  - hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
        type: c5.4xlarge
      zones:
        - cn-north-1a
      replicas: 3
  metadata:
    name: test-cluster 12
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 13
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    aws:
      region: cn-north-1 14
      propagateUserTags: true 15
      userTags:
        adminContact: jdoe
        costCenter: 7536
      subnets: 16
      - subnet-1
      - subnet-2
      - subnet-3
      amiID: ami-96c6f8f7 17 18
      serviceEndpoints: 19
      - name: ec2
        url: https://vpce-id.ec2.cn-north-1.vpce.amazonaws.com.cn
```

```

hostedZone: Z3URY6TWQ91KVV 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22
publish: Internal 23
pullSecret: '{"auths": ...}' 24

```

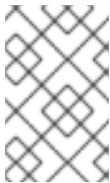
- 1 12 14 17 24** Required.
- 2** Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.
- 3 8 15** If you do not provide these parameters and values, the installation program provides the default value.
- 4** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 5 9** Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 6 10** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.
- 7 11** Whether to require the [Amazon EC2 Instance Metadata Service v2](#) (IMDSv2). To require IMDSv2, set the parameter value to **Required**. To allow the use of both IMDSv1 and IMDSv2, set the parameter value to **Optional**. If no value is specified, both IMDSv1 and IMDSv2 are allowed.



NOTE

The IMDS configuration for control plane machines that is set during cluster installation can only be changed by using the AWS CLI. The IMDS configuration for compute machines can be changed by using compute machine sets.

- 13** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 16** If you provide your own VPC, specify subnets for each availability zone that your cluster uses.
- 18** The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

- 19 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.
- 20 The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.
- 21 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 22 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 23 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

6.12.9.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.45. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.12.9.4. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.27. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***

- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.12.9.5. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.28. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

6.12.9.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.12.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$. /openshift-install create cluster --dir <installation_directory> \ 1  
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

- Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.12.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

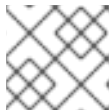
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.12.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.12.13. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

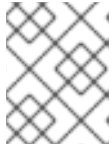
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

6.12.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.
- See [About remote health monitoring](#) for more information about the Telemetry service.

6.12.15. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

6.13. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN AWS BY USING CLOUDFORMATION TEMPLATES

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) that uses infrastructure that you provide.

One way to create this infrastructure is to use the provided CloudFormation templates. You can modify the templates to customize your infrastructure or use the information that they contain to create AWS objects according to your company's policies.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several CloudFormation templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

6.13.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or UNIX\)](#) in the AWS documentation.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.13.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.13.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

6.13.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 6.46. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

6.13.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.47. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

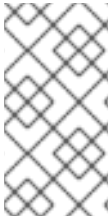
If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.13.3.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.29. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.13.3.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.30. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

6.13.3.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

6.13.4. Required AWS infrastructure components

To install OpenShift Container Platform on user-provisioned infrastructure in Amazon Web Services (AWS), you must manually create both the machines and their supporting infrastructure.

For more information about the integration testing for different platforms, see the [OpenShift Container Platform 4.x Tested Integrations](#) page.

By using the provided CloudFormation templates, you can create stacks of AWS resources that represent the following components:

- An AWS Virtual Private Cloud (VPC)
- Networking and load balancing components
- Security groups and roles
- An OpenShift Container Platform bootstrap node
- OpenShift Container Platform control plane nodes
- An OpenShift Container Platform compute node

Alternatively, you can manually create the components or you can reuse existing infrastructure that meets the cluster requirements. Review the CloudFormation templates for more details about how the components interrelate.

6.13.4.1. Other infrastructure components

- A VPC
- DNS entries
- Load balancers (classic or network) and listeners
- A public and a private Route 53 zone
- Security groups

- IAM roles
- S3 buckets

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.

Component	AWS type	Description	
Public subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.	
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.	
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkACL ● AWS::EC2::NetworkACLEntry 	You must allow the VPC to access the following ports:	
		Port	Reason
		80	Inbound HTTP traffic
		443	Inbound HTTPS traffic
		22	Inbound SSH traffic
		1024 - 65535	Inbound ephemeral traffic
	0 - 65535	Outbound ephemeral traffic	
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.	

Required DNS and load balancing components

Your DNS and load balancer configuration needs to use a public hosted zone and can use a private hosted zone similar to the one that the installation program uses if it provisions the cluster's infrastructure. You must create a DNS entry that resolves to your load balancer. An entry for **api.<cluster_name>.<domain>** must point to the external load balancer, and an entry for **api-int.<cluster_name>.<domain>** must point to the internal load balancer.

The cluster also requires load balancers and listeners for port 6443, which are required for the Kubernetes API and its extensions, and port 22623, which are required for the Ignition config files for new machines. The targets will be the control plane nodes. Port 6443 must be accessible to both clients external to the cluster and nodes within the cluster. Port 22623 must be accessible to nodes within the cluster.

Component	AWS type	Description
DNS	AWS::Route53::HostedZone	The hosted zone for your internal DNS.
Public load balancer	AWS::ElasticLoadBalancingV2::LoadBalancer	The load balancer for your public subnets.
External API server record	AWS::Route53::RecordSetGroup	Alias records for the external API server.
External listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 6443 for the external load balancer.
External target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the external load balancer.
Private load balancer	AWS::ElasticLoadBalancingV2::LoadBalancer	The load balancer for your private subnets.
Internal API server record	AWS::Route53::RecordSetGroup	Alias records for the internal API server.
Internal listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 22623 for the internal load balancer.

Component	AWS type	Description
Internal target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the internal load balancer.
Internal listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 6443 for the internal load balancer.
Internal target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the internal load balancer.

Security groups

The control plane and worker machines require access to the following ports:

Group	Type	IP Protocol	Port range
MasterSecurityGroup	AWS::EC2::SecurityGroup	icmp	0
		tcp	22
		tcp	6443
		tcp	22623
WorkerSecurityGroup	AWS::EC2::SecurityGroup	icmp	0
		tcp	22
BootstrapSecurityGroup	AWS::EC2::SecurityGroup	tcp	22
		tcp	19531

Control plane Ingress

The control plane machines require the following Ingress groups. Each Ingress group is a **AWS::EC2::SecurityGroupIngress** resource.

Ingress group	Description	IP protocol	Port range
MasterIngressEtcd	etcd	tcp	2379- 2380

Ingress group	Description	IP protocol	Port range
MasterIngress Vxlan	Vxlan packets	udp	4789
MasterIngress WorkerVxlan	Vxlan packets	udp	4789
MasterIngress Internal	Internal cluster communication and Kubernetes proxy metrics	tcp	9000 - 9999
MasterIngress WorkerInternal	Internal cluster communication	tcp	9000 - 9999
MasterIngress Kube	Kubernetes kubelet, scheduler and controller manager	tcp	10250 - 10259
MasterIngress WorkerKube	Kubernetes kubelet, scheduler and controller manager	tcp	10250 - 10259
MasterIngress IngressServices	Kubernetes Ingress services	tcp	30000 - 32767
MasterIngress WorkerIngressServices	Kubernetes Ingress services	tcp	30000 - 32767
MasterIngress Geneve	Geneve packets	udp	6081
MasterIngress WorkerGeneve	Geneve packets	udp	6081
MasterIngress IpsecIke	IPsec IKE packets	udp	500
MasterIngress WorkerIpsecIke	IPsec IKE packets	udp	500
MasterIngress IpsecNat	IPsec NAT-T packets	udp	4500

Ingress group	Description	IP protocol	Port range
MasterIngress WorkerIpsecNat	IPsec NAT-T packets	udp	4500
MasterIngress IpsecEsp	IPsec ESP packets	50	All
MasterIngress WorkerIpsecEsp	IPsec ESP packets	50	All
MasterIngress InternalUDP	Internal cluster communication	udp	9000 - 9999
MasterIngress WorkerInternalUDP	Internal cluster communication	udp	9000 - 9999
MasterIngress IngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767
MasterIngress WorkerIngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767

Worker Ingress

The worker machines require the following Ingress groups. Each Ingress group is a **AWS::EC2::SecurityGroupIngress** resource.

Ingress group	Description	IP protocol	Port range
WorkerIngress Vxlan	Vxlan packets	udp	4789
WorkerIngress WorkerVxlan	Vxlan packets	udp	4789
WorkerIngress Internal	Internal cluster communication	tcp	9000 - 9999
WorkerIngress WorkerInternal	Internal cluster communication	tcp	9000 - 9999

Ingress group	Description	IP protocol	Port range
WorkerIngress Kube	Kubernetes kubelet, scheduler, and controller manager	tcp	10250
WorkerIngress WorkerKube	Kubernetes kubelet, scheduler, and controller manager	tcp	10250
WorkerIngress IngressServices	Kubernetes Ingress services	tcp	30000 - 32767
WorkerIngress WorkerIngress Services	Kubernetes Ingress services	tcp	30000 - 32767
WorkerIngress Geneve	Geneve packets	udp	6081
WorkerIngress MasterGeneve	Geneve packets	udp	6081
WorkerIngress IpsecIke	IPsec IKE packets	udp	500
WorkerIngress MasterIpsecIke	IPsec IKE packets	udp	500
WorkerIngress IpsecNat	IPsec NAT-T packets	udp	4500
WorkerIngress MasterIpsecNat	IPsec NAT-T packets	udp	4500
WorkerIngress IpsecEsp	IPsec ESP packets	50	All
WorkerIngress MasterIpsecEsp	IPsec ESP packets	50	All
WorkerIngress InternalUDP	Internal cluster communication	udp	9000 - 9999
WorkerIngress MasterInternalUDP	Internal cluster communication	udp	9000 - 9999

Ingress group	Description	IP protocol	Port range
WorkerIngressIngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767
WorkerIngressMasterIngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767

Roles and instance profiles

You must grant the machines permissions in AWS. The provided CloudFormation templates grant the machines **Allow** permissions for the following **AWS::IAM::Role** objects and provide a **AWS::IAM::InstanceProfile** for each set of roles. If you do not use the templates, you can grant the machines the following broad permissions or the following individual permissions.

Role	Effect	Action	Resource
Master	Allow	ec2:*	*
	Allow	elasticloadbalancing:*	*
	Allow	iam:PassRole	*
	Allow	s3:GetObject	*
Worker	Allow	ec2:Describe*	*
Bootstrap	Allow	ec2:Describe*	*
	Allow	ec2:AttachVolume	*
	Allow	ec2:DetachVolume	*

6.13.4.2. Cluster machines

You need **AWS::EC2::Instance** objects for the following machines:

- A bootstrap machine. This machine is required during installation, but you can remove it after your cluster deploys.
- Three control plane machines. The control plane machines are not governed by a control plane machine set.
- Compute machines. You must create at least two compute machines, which are also known as worker machines, during installation. These machines are not governed by a compute machine set.

6.13.4.3. Required AWS permissions for the IAM user



NOTE

Your IAM user must have the permission **tag:GetResources** in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

When you attach the **AdministratorAccess** policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

Example 6.31. Required EC2 permissions for installation

- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteSnapshot**
- **ec2>DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInstanceTypes**
- **ec2:DescribeInternetGateways**

- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

Example 6.32. Required permissions for creating network resources during installation

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**

- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



NOTE

If you use an existing VPC, your account does not require these permissions for creating network resources.

Example 6.33. Required Elastic Load Balancing permissions (ELB) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

Example 6.34. Required Elastic Load Balancing permissions (ELBv2) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

Example 6.35. Required IAM permissions for installation

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam>DeleteRole**
- **iam>DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**

- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**



NOTE

If you have not created a load balancer in your AWS account, the IAM user also requires the **iam:CreateServiceLinkedRole** permission.

Example 6.36. Required Route 53 permissions for installation

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53>DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

Example 6.37. Required S3 permissions for installation

- **s3:CreateBucket**
- **s3>DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**

- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketPolicy**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

Example 6.38. S3 permissions that cluster Operators require

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

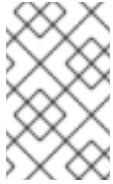
Example 6.39. Required permissions to delete base cluster resources

- **autoscaling:DescribeAutoScalingGroups**

- **ec2:DeletePlacementGroup**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam>ListAttachedRolePolicies**
- **iam>ListInstanceProfiles**
- **iam>ListRolePolicies**
- **iam>ListUserPolicies**
- **s3>DeleteObject**
- **s3>ListBucketVersions**
- **tag:GetResources**

Example 6.40. Required permissions to delete network resources

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **ec2:ReplaceRouteTableAssociation**

**NOTE**

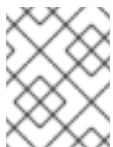
If you use an existing VPC, your account does not require these permissions to delete network resources. Instead, your account only requires the **tag:UntagResources** permission to delete network resources.

Example 6.41. Required permissions to delete a cluster with shared instance roles

- **iam:UntagRole**

Example 6.42. Additional IAM and S3 permissions that are required to create manifests

- **iam>DeleteAccessKey**
- **iam>DeleteUser**
- **iam>DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam>ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:ListBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

**NOTE**

If you are managing your cloud provider credentials with mint mode, the IAM user also requires the **iam:CreateAccessKey** and **iam:CreateUser** permissions.

Example 6.43. Optional permissions for instance and quota checks for installation

- **ec2:DescribeInstanceTypeOfferings**
- **servicequotas:ListAWSDefaultServiceQuotas**

6.13.5. Obtaining an AWS Marketplace image

If you are deploying an OpenShift Container Platform cluster using an AWS Marketplace image, you must first subscribe through AWS. Subscribing to the offer provides you with the AMI ID that the installation program uses to deploy worker nodes.

Prerequisites

- You have an AWS account to purchase the offer. This account does not have to be the same account that is used to install the cluster.

Procedure

1. Complete the OpenShift Container Platform subscription from the [AWS Marketplace](#).
2. Record the AMI ID for your specific region. If you use the CloudFormation template to deploy your worker nodes, you must update the **worker0.type.properties.ImageID** parameter with this value.

6.13.6. Obtaining the installation program

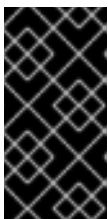
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

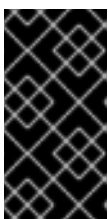
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

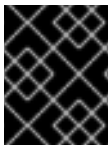
- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.13.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the `core` user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user `core`. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

- If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

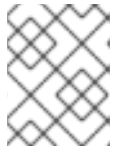
- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

6.13.8. Creating the installation files for AWS

To install OpenShift Container Platform on Amazon Web Services (AWS) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and

customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

6.13.8.1. Optional: Creating a separate **/var** partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

- Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

- Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> ❶
      partitions:
        - label: var
          start_mib: <partition_start_offset> ❷
          size_mib: <partition_size> ❸
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] ❹
          with_mount_unit: true
```

- The storage device name of the disk that you want to partition.
- When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- The size of the data partition in mebibytes.
- The **prjquota** mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

6.13.8.2. Creating the installation configuration file

Generate and customize the installation configuration file that the installation program needs to deploy your cluster.

Prerequisites

- You obtained the OpenShift Container Platform installation program for user-provisioned infrastructure and the pull secret for your cluster.
- You checked that you are deploying your cluster to a region with an accompanying Red Hat Enterprise Linux CoreOS (RHCOS) AMI published by Red Hat. If you are deploying to a region that requires a custom AMI, such as an AWS GovCloud region, you must create the **install-config.yaml** file manually.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.



IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **aws** as the platform to target.
- iii. If you do not have an AWS profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.



NOTE

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

- iv. Select the AWS region to deploy the cluster to.
 - v. Select the base domain for the Route 53 service that you configured for your cluster.
 - vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Optional: Back up the **install-config.yaml** file.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- See [Configuration and credential file settings](#) in the AWS documentation for more information about AWS profile and credential configuration.

6.13.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

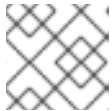
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle**

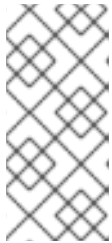
in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.13.8.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the control plane machine set:

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

```
1. Open the <installation_directory>/manifests/cluster-scheduler-02-config.yml file
```

- a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
 - c. Save and exit the file.
5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: ❶
    id: mycluster-100419-private-zone
  publicZone: ❷
    id: example.openshift.com
status: {}
```

- ❶ ❷ Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. Optional: If you manually created a cloud identity and access management (IAM) role, locate any `CredentialsRequest` objects with the `TechPreviewNoUpgrade` annotation in the release image by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=<platform_name>
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```



IMPORTANT

The release image includes `CredentialsRequest` objects for Technology Preview features that are enabled by the `TechPreviewNoUpgrade` feature set. You can identify these objects by their use of the `release.openshift.io/feature-set: TechPreviewNoUpgrade` annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- a. Delete all `CredentialsRequest` objects that have the `TechPreviewNoUpgrade` annotation.

- To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

6.13.9. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Amazon Web Services (AWS). The infrastructure name is also used to locate the appropriate AWS resources during an OpenShift Container Platform installation. The provided CloudFormation templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- The output of this command is your cluster name and a random string.

6.13.10. Creating a VPC in AWS

You must create a Virtual Private Cloud (VPC) in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements, including VPN and route tables.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the VPC.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.

Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

- 1** The CIDR block for the VPC.
- 2** Specify a CIDR block in the format **x.x.x.x/16-24**.
- 3** The number of availability zones to deploy the VPC in.
- 4** Specify an integer between **1** and **3**.
- 5** The size of each subnet in each availability zone.
- 6** Specify an integer between **5** and **13**, where **5** is **/27** and **13** is **/19**.

- Copy the template from the **CloudFormation template for the VPC** section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.
- Launch the CloudFormation template to create a stack of AWS resources that represent the VPC:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
```

- <name>** is the name for the CloudFormation stack, such as **cluster-vpc**. You need the name of this stack if you remove the cluster.
- <template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- <parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

- Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

VpcId	The ID of your VPC.
PublicSubnetIds	The IDs of the new public subnets.
PrivateSubnetIds	The IDs of the new private subnets.

6.13.10.1. CloudFormation template for the VPC

You can use the following CloudFormation template to deploy the VPC that you need for your OpenShift Container Platform cluster.

Example 6.44. CloudFormation template for the VPC

AWSTemplateFormatVersion: 2010-09-09

Description: Template for Best Practice VPC with 1-3 AZs

Parameters:

VpcCidr:

AllowedPattern: ^(((0-9)|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5]).){3}([0-9]|1[0-9]|2[0-4][0-9]|25[0-5])(\.(1[6-9]|2[0-4]))\$

ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.

Default: 10.0.0/16

Description: CIDR block for VPC.

Type: String

AvailabilityZoneCount:

ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"

MinValue: 1

MaxValue: 3

Default: 1

Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"

Type: Number

SubnetBits:

ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.

MinValue: 5

MaxValue: 13

Default: 12

Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"

Type: Number

Metadata:

AWS::CloudFormation::Interface:

ParameterGroups:

- Label:

default: "Network Configuration"

Parameters:

- VpcCidr

- SubnetBits

- Label:

default: "Availability Zones"

Parameters:

- AvailabilityZoneCount

ParameterLabels:

AvailabilityZoneCount:

default: "Availability Zone Count"

VpcCidr:

default: "VPC CIDR"

SubnetBits:

default: "Bits Per Subnet"

Conditions:

DoAz3: !Equals [3, !Ref AvailabilityZoneCount]

DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:

VPC:

Type: "AWS::EC2::VPC"

```
Properties:
  EnableDnsSupport: "true"
  EnableDnsHostnames: "true"
  CidrBlock: !Ref VpcCidr
PublicSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
InternetGateway:
  Type: "AWS::EC2::InternetGateway"
GatewayToInternet:
  Type: "AWS::EC2::VPCGatewayAttachment"
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref InternetGateway
PublicRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PublicRoute:
  Type: "AWS::EC2::Route"
  DependsOn: GatewayToInternet
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
```

```

Properties:
  SubnetId: !Ref PublicSubnet2
  RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
  Condition: DoAz3
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP
        - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 1

```

```
- Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
```

```

RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
      - Effect: Allow
        Principal: '*'
        Action:
          - '*'
        Resource:
          - '*'
    RouteTableIds:
    - !Ref PublicRouteTable
    - !Ref PrivateRouteTable
    - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
    - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
    ServiceName: !Join
      - "
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
    Vpclid: !Ref VPC

Outputs:
Vpclid:
  Description: ID of the new VPC.
  Value: !Ref VPC
PublicSubnetIds:
  Description: Subnet IDs of the public subnets.

```

```

Value:
  !Join [
    "",
    [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
  ]
PrivateSubnetIds:
  Description: Subnet IDs of the private subnets.
  Value:
    !Join [
      "",
      [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
    ]

```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).

6.13.11. Creating networking and load balancing components in AWS

You must configure networking and classic or network load balancing in Amazon Web Services (AWS) that your OpenShift Container Platform cluster can use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the networking and load balancing components that your OpenShift Container Platform cluster requires. The template also creates a hosted zone and subnet tags.

You can run the template multiple times within a single Virtual Private Cloud (VPC).



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

Procedure

- Obtain the hosted zone ID for the Route 53 base domain that you specified in the **install-config.yaml** file for your cluster. You can obtain details about your hosted zone by running the following command:

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

- 1** For the `<route53_domain>`, specify the Route 53 base domain that you used when you generated the `install-config.yaml` file for the cluster.

Example output

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

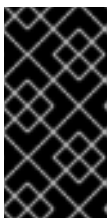
In the example output, the hosted zone ID is **Z21IXYZABCZ2A4**.

- Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

- 1** A short, representative cluster name to use for hostnames, etc.
- 2** Specify the cluster name that you used when you generated the `install-config.yaml` file for the cluster.
- 3** The name for your cluster infrastructure that is encoded in your Ignition config files for the

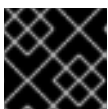
- 4 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
 - 5 The Route 53 public zone ID to register the targets with.
 - 6 Specify the Route 53 public zone ID, which as a format similar to **Z21IXYZABCZ2A4**. You can obtain this value from the AWS console.
 - 7 The Route 53 zone to register the targets with.
 - 8 Specify the Route 53 base domain that you used when you generated the **install-config.yaml** file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
 - 9 The public subnets that you created for your VPC.
 - 10 Specify the **PublicSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 11 The private subnets that you created for your VPC.
 - 12 Specify the **PrivateSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 13 The VPC that you created for the cluster.
 - 14 Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
3. Copy the template from the **CloudFormation template for the network and load balancers** section of this topic and save it as a YAML file on your computer. This template describes the networking and load balancing objects that your cluster requires.



IMPORTANT

If you are deploying your cluster to an AWS government or secret region, you must update the **InternalApiServerRecord** in the CloudFormation template to use **CNAME** records. Records of type **ALIAS** are not supported for AWS government regions.

4. Launch the CloudFormation template to create a stack of AWS resources that provide the networking and load balancing components:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-dns**. You need the name of this stack if you remove the cluster.

- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.
- 4 You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

PrivateHostedZoneId	Hosted zone ID for the private DNS.
ExternalApiLoadBalancerName	Full name of the external API load balancer.
InternalApiLoadBalancerName	Full name of the internal API load balancer.
ApiServerDnsName	Full hostname of the API server.
RegisterNlbTargetLambda	Lambda ARN useful to help register/deregister IP targets for these load balancers.
ExternalApiTargetGroupArn	ARN of external API target group.
InternalApiTargetGroupArn	ARN of internal API target group.

InternalServiceTargetGroupArn

ARN of internal service target group.

6.13.11.1. CloudFormation template for the network and load balancers

You can use the following CloudFormation template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster.

Example 6.45. CloudFormation template for the network and load balancers

AWSTemplateFormatVersion: [2010-09-09](#)

Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:

ClusterName:

AllowedPattern: `^[a-zA-Z][a-zA-Z0-9\-\-]{0,26}$`

MaxLength: [27](#)

MinLength: [1](#)

ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a maximum of [27](#) characters.

Description: A short, representative cluster name to use for host names and other identifying names.

Type: String

InfrastructureName:

AllowedPattern: `^[a-zA-Z][a-zA-Z0-9\-\-]{0,26}$`

MaxLength: [27](#)

MinLength: [1](#)

ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of [27](#) characters.

Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.

Type: String

HostedZoneId:

Description: The Route53 public zone ID to register the targets with, such as Z21IXYZABCZ2A4.

Type: String

HostedZoneName:

Description: The Route53 zone to register the targets with, such as example.com. Omit the trailing period.

Type: String

Default: "example.com"

PublicSubnets:

Description: The internet-facing subnets.

Type: List<AWS::EC2::Subnet::Id>

PrivateSubnets:

Description: The internal subnets.

Type: List<AWS::EC2::Subnet::Id>

VpcId:

Description: The VPC-scoped resources will belong to this VPC.

Type: AWS::EC2::VPC::Id

Metadata:

```

AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label:
    default: "Cluster Information"
    Parameters:
    - ClusterName
    - InfrastructureName
  - Label:
    default: "Network Configuration"
    Parameters:
    - VpcId
    - PublicSubnets
    - PrivateSubnets
  - Label:
    default: "DNS"
    Parameters:
    - HostedZoneName
    - HostedZoneId
  ParameterLabels:
  ClusterName:
    default: "Cluster Name"
  InfrastructureName:
    default: "Infrastructure Name"
  VpcId:
    default: "VPC ID"
  PublicSubnets:
    default: "Public Subnets"
  PrivateSubnets:
    default: "Private Subnets"
  HostedZoneName:
    default: "Public Hosted Zone Name"
  HostedZoneId:
    default: "Public Hosted Zone ID"

```

```

Resources:
  ExtApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
      IpAddressType: ipv4
      Subnets: !Ref PublicSubnets
      Type: network

```

```

  IntApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "int"]]
      Scheme: internal
      IpAddressType: ipv4
      Subnets: !Ref PrivateSubnets
      Type: network

```

```

  IntDns:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:

```

```

    Comment: "Managed by CloudFormation"
    Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
    HostedZoneTags:
    - Key: Name
      Value: !Join [ "-", [!Ref InfrastructureName, "int"]]
    - Key: !Join [ "", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
    VPCs:
    - VPCId: !Ref Vpclid
      VPCRegion: !Ref "AWS::Region"

```

ExternalApiServerRecord:

```

Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref HostedZoneId
  RecordSets:
  - Name:
    !Join [
      ".",
      ["api", !Ref ClusterName, !Join [ "", [!Ref HostedZoneName, "."]]],
    ]
    Type: A
    AliasTarget:
      HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
      DNSName: !GetAtt ExtApiElb.DNSName

```

InternalApiServerRecord:

```

Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref IntDns
  RecordSets:
  - Name:
    !Join [
      ".",
      ["api", !Ref ClusterName, !Join [ "", [!Ref HostedZoneName, "."]]],
    ]
    Type: A
    AliasTarget:
      HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
      DNSName: !GetAtt IntApiElb.DNSName
  - Name:
    !Join [
      ".",
      ["api-int", !Ref ClusterName, !Join [ "", [!Ref HostedZoneName, "."]]],
    ]
    Type: A
    AliasTarget:
      HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
      DNSName: !GetAtt IntApiElb.DNSName

```

ExternalApiListener:

```

Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
  DefaultActions:

```

- Type: forward
TargetGroupArn:
Ref: ExternalApiTargetGroup
LoadBalancerArn:
Ref: ExtApiElb
Port: 6443
Protocol: TCP

ExternalApiTargetGroup:
Type: AWS::ElasticLoadBalancingV2::TargetGroup
Properties:
HealthCheckIntervalSeconds: 10
HealthCheckPath: "/readyz"
HealthCheckPort: 6443
HealthCheckProtocol: HTTPS
HealthyThresholdCount: 2
UnhealthyThresholdCount: 2
Port: 6443
Protocol: TCP
TargetType: ip
VpcId:
Ref: VpcId
TargetGroupAttributes:
- Key: deregistration_delay.timeout_seconds
Value: 60

InternalApiListener:
Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
DefaultActions:
- Type: forward
TargetGroupArn:
Ref: InternalApiTargetGroup
LoadBalancerArn:
Ref: IntApiElb
Port: 6443
Protocol: TCP

InternalApiTargetGroup:
Type: AWS::ElasticLoadBalancingV2::TargetGroup
Properties:
HealthCheckIntervalSeconds: 10
HealthCheckPath: "/readyz"
HealthCheckPort: 6443
HealthCheckProtocol: HTTPS
HealthyThresholdCount: 2
UnhealthyThresholdCount: 2
Port: 6443
Protocol: TCP
TargetType: ip
VpcId:
Ref: VpcId
TargetGroupAttributes:
- Key: deregistration_delay.timeout_seconds
Value: 60

InternalServiceInternalListener:
 Type: AWS::ElasticLoadBalancingV2::Listener
 Properties:
 DefaultActions:
 - Type: forward
 TargetGroupArn:
 Ref: InternalServiceTargetGroup
 LoadBalancerArn:
 Ref: IntApiElb
 Port: 22623
 Protocol: TCP

InternalServiceTargetGroup:
 Type: AWS::ElasticLoadBalancingV2::TargetGroup
 Properties:
 HealthCheckIntervalSeconds: 10
 HealthCheckPath: "/healthz"
 HealthCheckPort: 22623
 HealthCheckProtocol: HTTPS
 HealthyThresholdCount: 2
 UnhealthyThresholdCount: 2
 Port: 22623
 Protocol: TCP
 TargetType: ip
 Vpclid:
 Ref: Vpclid
 TargetGroupAttributes:
 - Key: deregistration_delay.timeout_seconds
 Value: 60

RegisterTargetLambdalamRole:
 Type: AWS::IAM::Role
 Properties:
 RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
 AssumeRolePolicyDocument:
 Version: "2012-10-17"
 Statement:
 - Effect: "Allow"
 Principal:
 Service:
 - "lambda.amazonaws.com"
 Action:
 - "sts:AssumeRole"
 Path: "/"
 Policies:
 - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
 PolicyDocument:
 Version: "2012-10-17"
 Statement:
 - Effect: "Allow"
 Action:
 [
 "elasticloadbalancing:RegisterTargets",
 "elasticloadbalancing:DeregisterTargets",
]
 Resource: !Ref InternalApiTargetGroup

```

- Effect: "Allow"
  Action:
  [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
  ]
  Resource: !Ref InternalServiceTargetGroup
- Effect: "Allow"
  Action:
  [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
  ]
  Resource: !Ref ExternalApiTargetGroup

```

RegisterNlbIpTargets:

```

Type: "AWS::Lambda::Function"
Properties:
  Handler: "index.handler"
  Role:
    Fn::GetAtt:
    - "RegisterTargetLambdalamRole"
    - "Arn"
  Code:
    ZipFile: |
      import json
      import boto3
      import cfnresponse
      def handler(event, context):
        elb = boto3.client('elbv2')
        if event['RequestType'] == 'Delete':
          elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
[TargetArn],Targets=[{'Id': event['ResourceProperties']['TargetIp']})
          elif event['RequestType'] == 'Create':
            elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetIp']})
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
  Runtime: "python3.8"
  Timeout: 120

```

RegisterSubnetTagsLambdalamRole:

```

Type: AWS::IAM::Role
Properties:
  RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
  AssumeRolePolicyDocument:
    Version: "2012-10-17"
    Statement:
    - Effect: "Allow"
      Principal:
        Service:
        - "lambda.amazonaws.com"
      Action:
      - "sts:AssumeRole"
  Path: "/"

```

Policies:

```
- PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
```

PolicyDocument:

```
Version: "2012-10-17"
```

Statement:

```
- Effect: "Allow"
```

Action:

```
[
  "ec2:DeleteTags",
  "ec2:CreateTags"
]
```

```
Resource: "arn:aws:ec2:*:*:subnet/*"
```

```
- Effect: "Allow"
```

Action:

```
[
  "ec2:DescribeSubnets",
  "ec2:DescribeTags"
]
```

```
Resource: ""
```

RegisterSubnetTags:

```
Type: "AWS::Lambda::Function"
```

Properties:

```
Handler: "index.handler"
```

Role:

```
Fn::GetAtt:
```

```
- "RegisterSubnetTagsLambdalamRole"
```

```
- "Arn"
```

Code:

```
ZipFile: |
```

```
import json
```

```
import boto3
```

```
import cfnresponse
```

```
def handler(event, context):
```

```
    ec2_client = boto3.client('ec2')
```

```
    if event['RequestType'] == 'Delete':
```

```
        for subnet_id in event['ResourceProperties']['Subnets']:
```

```
            ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
```

```
    elif event['RequestType'] == 'Create':
```

```
        for subnet_id in event['ResourceProperties']['Subnets']:
```

```
            ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
```

```
        responseData = {}
```

```
        cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
```

```
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
```

```
Runtime: "python3.8"
```

```
Timeout: 120
```

RegisterPublicSubnetTags:

```
Type: Custom::SubnetRegister
```

Properties:

```
ServiceToken: !GetAtt RegisterSubnetTags.Arn
```

```
InfrastructureName: !Ref InfrastructureName
```

```
Subnets: !Ref PublicSubnets
```


RegisterPrivateSubnetTags:
 Type: Custom::SubnetRegister
 Properties:
 ServiceToken: !GetAtt RegisterSubnetTags.Arn
 InfrastructureName: !Ref InfrastructureName
 Subnets: !Ref PrivateSubnets

Outputs:

PrivateHostedZoneId:
 Description: Hosted zone ID for the private DNS, which is required for private records.
 Value: !Ref IntDns

ExternalApiLoadBalancerName:
 Description: Full name of the external API load balancer.
 Value: !GetAtt ExtApiElb.LoadBalancerFullName

InternalApiLoadBalancerName:
 Description: Full name of the internal API load balancer.
 Value: !GetAtt IntApiElb.LoadBalancerFullName

ApiServerDnsName:
 Description: Full hostname of the API server, which is required for the Ignition config files.
 Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]

RegisterNlbTargetsLambda:
 Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.
 Value: !GetAtt RegisterNlbTargets.Arn

ExternalApiTargetGroupArn:
 Description: ARN of the external API target group.
 Value: !Ref ExternalApiTargetGroup

InternalApiTargetGroupArn:
 Description: ARN of the internal API target group.
 Value: !Ref InternalApiTargetGroup

InternalServiceTargetGroupArn:
 Description: ARN of the internal service target group.
 Value: !Ref InternalServiceTargetGroup

IMPORTANT

If you are deploying your cluster to an AWS government or secret region, you must update the **InternalApiServerRecord** to use **CNAME** records. Records of type **ALIAS** are not supported for AWS government regions. For example:

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).
- You can view details about your hosted zones by navigating to the [AWS Route 53 console](#).

- See [Listing public hosted zones](#) in the AWS documentation for more information about listing public hosted zones.

6.13.12. Creating security group and roles in AWS

You must create security groups and roles in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the security groups and roles that your OpenShift Container Platform cluster requires.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

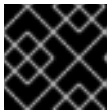
Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

- 1** The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

- 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
 - 3 The CIDR block for the VPC.
 - 4 Specify the CIDR block parameter that you used for the VPC that you defined in the form **x.x.x.x/16-24**.
 - 5 The private subnets that you created for your VPC.
 - 6 Specify the **PrivateSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 7 The VPC that you created for the cluster.
 - 8 Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
2. Copy the template from the **CloudFormation template for security objects** section of this topic and save it as a YAML file on your computer. This template describes the security groups and roles that your cluster requires.
 3. Launch the CloudFormation template to create a stack of AWS resources that represent the security groups and roles:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-sec**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.
- 4 You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** and **AWS::IAM::InstanceProfile** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. Confirm that the template components exist:

■

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

MasterSecurityGroupID	Master Security Group ID
WorkerSecurityGroupID	Worker Security Group ID
MasterInstanceProfile	Master IAM Instance Profile
WorkerInstanceProfile	Worker IAM Instance Profile

6.13.12.1. CloudFormation template for security objects

You can use the following CloudFormation template to deploy the security objects that you need for your OpenShift Container Platform cluster.

Example 6.46. CloudFormation template for security objects

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\_]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
    maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
    used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(((0-9){1,9}[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.)\.{3}((0-9){1,9}[0-9]|1[0-9]{2}|2[0-
    4][0-9]|25[0-5])\.(1[6-9]|2[0-4])$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
```

Description: The internal subnets.
 Type: List<AWS::EC2::Subnet::Id>

Metadata:

AWS::CloudFormation::Interface:

ParameterGroups:

- Label:

default: "Cluster Information"

Parameters:

- InfrastructureName

- Label:

default: "Network Configuration"

Parameters:

- VpcId

- VpcCidr

- PrivateSubnets

ParameterLabels:

InfrastructureName:

default: "Infrastructure Name"

VpcId:

default: "VPC ID"

VpcCidr:

default: "VPC CIDR"

PrivateSubnets:

default: "Private Subnets"

Resources:

MasterSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Cluster Master Security Group

SecurityGroupIngress:

- IpProtocol: icmp

FromPort: 0

ToPort: 0

CidrIp: !Ref VpcCidr

- IpProtocol: tcp

FromPort: 22

ToPort: 22

CidrIp: !Ref VpcCidr

- IpProtocol: tcp

ToPort: 6443

FromPort: 6443

CidrIp: !Ref VpcCidr

- IpProtocol: tcp

FromPort: 22623

ToPort: 22623

CidrIp: !Ref VpcCidr

VpcId: !Ref VpcId

WorkerSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Cluster Worker Security Group

SecurityGroupIngress:

- IpProtocol: icmp

FromPort: 0
ToPort: 0
CidrIp: !Ref VpcCidr
- IpProtocol: tcp
FromPort: 22
ToPort: 22
CidrIp: !Ref VpcCidr
Vpclid: !Ref Vpclid

MasterIngressEtcd:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: etcd
FromPort: 2379
ToPort: 2380
IpProtocol: tcp

MasterIngressVxlan:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Vxlan packets
FromPort: 4789
ToPort: 4789
IpProtocol: udp

MasterIngressWorkerVxlan:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Vxlan packets
FromPort: 4789
ToPort: 4789
IpProtocol: udp

MasterIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

MasterIngressWorkerGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081

ToPort: 6081
IpProtocol: udp

MasterIngressIpsecIke:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec IKE packets
FromPort: 500
ToPort: 500
IpProtocol: udp

MasterIngressIpsecNat:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec NAT-T packets
FromPort: 4500
ToPort: 4500
IpProtocol: udp

MasterIngressIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec ESP packets
IpProtocol: 50

MasterIngressWorkerIpsecIke:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: IPsec IKE packets
FromPort: 500
ToPort: 500
IpProtocol: udp

MasterIngressWorkerIpsecNat:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: IPsec NAT-T packets
FromPort: 4500
ToPort: 4500
IpProtocol: udp

MasterIngressWorkerIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec ESP packets
IpProtocol: 50

MasterIngressInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: tcp

MasterIngressWorkerInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: tcp

MasterIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

MasterIngressWorkerInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

MasterIngressKube:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes kubelet, scheduler and controller manager

FromPort: 10250

ToPort: 10259

IpProtocol: tcp

MasterIngressWorkerKube:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes kubelet, scheduler and controller manager
FromPort: 10250
ToPort: 10259
IpProtocol: tcp

MasterIngressIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

MasterIngressWorkerIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

MasterIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

WorkerIngressVxlan:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Vxlan packets
FromPort: 4789
ToPort: 4789
IpProtocol: udp

WorkerIngressMasterVxlan:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Vxlan packets

FromPort: 4789

ToPort: 4789

IpProtocol: udp

WorkerIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Geneve packets

FromPort: 6081

ToPort: 6081

IpProtocol: udp

WorkerIngressMasterGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Geneve packets

FromPort: 6081

ToPort: 6081

IpProtocol: udp

WorkerIngressIpsecIke:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec IKE packets

FromPort: 500

ToPort: 500

IpProtocol: udp

WorkerIngressIpsecNat:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec NAT-T packets

FromPort: 4500

ToPort: 4500

IpProtocol: udp

WorkerIngressIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec ESP packets
IpProtocol: 50

WorkerIngressMasterIpseclke:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec IKE packets
FromPort: 500
ToPort: 500
IpProtocol: udp

WorkerIngressMasterIpsecNat:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec NAT-T packets
FromPort: 4500
ToPort: 4500
IpProtocol: udp

WorkerIngressMasterIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: IPsec ESP packets
IpProtocol: 50

WorkerIngressInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

WorkerIngressMasterInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

WorkerIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

WorkerIngressMasterInternalUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
 Description: Internal cluster communication
 FromPort: 9000
 ToPort: 9999
 IpProtocol: udp

WorkerIngressKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
 Description: Kubernetes secure kubelet port
 FromPort: 10250
 ToPort: 10250
 IpProtocol: tcp

WorkerIngressWorkerKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
 Description: Internal Kubernetes communication
 FromPort: 10250
 ToPort: 10250
 IpProtocol: tcp

WorkerIngressIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
 Description: Kubernetes ingress services
 FromPort: 30000
 ToPort: 32767
 IpProtocol: tcp

WorkerIngressMasterIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
 Description: Kubernetes ingress services
 FromPort: 30000
 ToPort: 32767
 IpProtocol: tcp

WorkerIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

MasterIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action:

- "ec2:AttachVolume"

- "ec2:AuthorizeSecurityGroupIngress"

- "ec2:CreateSecurityGroup"

- "ec2:CreateTags"

- "ec2:CreateVolume"

- "ec2>DeleteSecurityGroup"

- "ec2>DeleteVolume"

- "ec2:Describe*"

- "ec2:DetachVolume"

- "ec2:ModifyInstanceAttribute"

- "ec2:ModifyVolume"

- "ec2:RevokeSecurityGroupIngress"

- "elasticloadbalancing:AddTags"

- "elasticloadbalancing:AttachLoadBalancerToSubnets"

- "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"

- "elasticloadbalancing:CreateListener"

- "elasticloadbalancing:CreateLoadBalancer"

- "elasticloadbalancing:CreateLoadBalancerPolicy"

- "elasticloadbalancing:CreateLoadBalancerListeners"
- "elasticloadbalancing:CreateTargetGroup"
- "elasticloadbalancing:ConfigureHealthCheck"
- "elasticloadbalancing>DeleteListener"
- "elasticloadbalancing>DeleteLoadBalancer"
- "elasticloadbalancing>DeleteLoadBalancerListeners"
- "elasticloadbalancing>DeleteTargetGroup"
- "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
- "elasticloadbalancing:DeregisterTargets"
- "elasticloadbalancing:Describe*"
- "elasticloadbalancing:DetachLoadBalancerFromSubnets"
- "elasticloadbalancing:ModifyListener"
- "elasticloadbalancing:ModifyLoadBalancerAttributes"
- "elasticloadbalancing:ModifyTargetGroup"
- "elasticloadbalancing:ModifyTargetGroupAttributes"
- "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
- "elasticloadbalancing:RegisterTargets"
- "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
- "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
- "kms:DescribeKey"

Resource: ""

MasterInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Roles:

- Ref: "MasterIamRole"

WorkerIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action:

- "ec2:DescribeInstances"

- "ec2:DescribeRegions"

Resource: ""

WorkerInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Roles:

- Ref: "WorkerIamRole"

Outputs:**MasterSecurityGroupId:**

Description: Master Security Group ID

Value: !GetAtt MasterSecurityGroup.GroupId

WorkerSecurityGroupId:

Description: Worker Security Group ID

Value: !GetAtt WorkerSecurityGroup.GroupId

MasterInstanceProfile:

Description: Master IAM Instance Profile

Value: !Ref MasterInstanceProfile

WorkerInstanceProfile:

Description: Worker IAM Instance Profile

Value: !Ref WorkerInstanceProfile

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).

6.13.13. Accessing RHCOS AMIs with stream metadata

In OpenShift Container Platform, *stream metadata* provides standardized metadata about RHCOS in the JSON format and injects the metadata into the cluster. Stream metadata is a stable format that supports multiple architectures and is intended to be self-documenting for maintaining automation.

You can use the **coreos print-stream-json** sub-command of **openshift-install** to access information about the boot images in the stream metadata format. This command provides a method for printing stream metadata in a scriptable, machine-readable format.

For user-provisioned installations, the **openshift-install** binary contains references to the version of RHCOS boot images that are tested for use with OpenShift Container Platform, such as the AWS AMI.

Procedure

To parse the stream metadata, use one of the following methods:

- From a Go program, use the official **stream-metadata-go** library at <https://github.com/coreos/stream-metadata-go>. You can also view example code in the library.
- From another programming language, such as Python or Ruby, use the JSON library of your preferred programming language.
- From a command-line utility that handles JSON data, such as **jq**:
 - Print the current **x86_64** or **aarch64** AMI for an AWS region, such as **us-west-1**:

For x86_64

```
$ openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.images.aws.regions["us-west-1"].image'
```

Example output

```
ami-0d3e625f84626bbda
```

For aarch64

```
$ openshift-install coreos print-stream-json | jq -r
'.architectures.aarch64.images.aws.regions["us-west-1"].image'
```

Example output

```
ami-0af1d3b7fa5be2131
```

The output of this command is the AWS AMI ID for your designated architecture and the **us-west-1** region. The AMI must belong to the same region as the cluster.

6.13.14. RHCOS AMIs for the AWS infrastructure

Red Hat provides Red Hat Enterprise Linux CoreOS (RHCOS) AMIs that are valid for the various AWS regions and instance architectures that you can manually specify for your OpenShift Container Platform nodes.

**NOTE**

By importing your own AMI, you can also install to regions that do not have a published RHCOS AMI.

Table 6.48. x86_64 RHCOS AMIs

AWS zone	AWS AMI
af-south-1	ami-073850a7021953a5c
ap-east-1	ami-0f8800a05c09be42d
ap-northeast-1	ami-0a226dbcc9a561c40
ap-northeast-2	ami-041ae0537e2eddec1
ap-northeast-3	ami-0bb8d9b69dc5b7670
ap-south-1	ami-0e9c18058fc5f94fd
ap-southeast-1	ami-03022d358ba2168be
ap-southeast-2	ami-09ffdc5be9b973be0
ap-southeast-3	ami-0facf1a0edeb20314

AWS zone	AWS AMI
ca-central-1	ami-028cea206c2d03317
eu-central-1	ami-002eb441f329ccb0f
eu-north-1	ami-0b1a1fb68b3b9fee7
eu-south-1	ami-0bd0fd41a1d3f799a
eu-west-1	ami-04504e8799057980c
eu-west-2	ami-0cc9297ddb3bce971
eu-west-3	ami-06f98f607a50937c6
me-south-1	ami-0fe39da7871a5b2a5
sa-east-1	ami-08265cc3226697767
us-east-1	ami-0fe05b1aa8dacfa90
us-east-2	ami-0ff64f495c7e977cf
us-gov-east-1	ami-0c99658076c41872a
us-gov-west-1	ami-0ca4acd5b8ba1cb1d
us-west-1	ami-01dc5d8e6bb6f23f4
us-west-2	ami-0404a109adfd00019

Table 6.49. aarch64 RHCOS AMIs

AWS zone	AWS AMI
af-south-1	ami-0574bcc5f80b0ad9a
ap-east-1	ami-0a65e79822ae2d235
ap-northeast-1	ami-0f7ef19d48e22353b
ap-northeast-2	ami-051dc6de359975e3c
ap-northeast-3	ami-0fd0b4222595650ac

AWS zone	AWS AMI
ap-south-1	ami-05f9d14fe4a90ed6f
ap-southeast-1	ami-0afdb9133d22fba5f
ap-southeast-2	ami-0ef979abe82d07d44
ap-southeast-3	ami-025f9103ac4310e7f
ca-central-1	ami-0588cdf59e5c14847
eu-central-1	ami-0ef24c0e18f93fa42
eu-north-1	ami-0439e2a3bf315df1a
eu-south-1	ami-0714e7c2e0106cdd3
eu-west-1	ami-0b960e76764ccd0c3
eu-west-2	ami-02621f50de62b3b89
eu-west-3	ami-0933ce7f5e2bfb50e
me-south-1	ami-074bde61a2ab740ee
sa-east-1	ami-03b4f97cfc8033ae0
us-east-1	ami-02a574449d4f4d280
us-east-2	ami-020e5600ef28c60ae
us-gov-east-1	ami-069f60e1dcf766d24
us-gov-west-1	ami-0db3cda4dbaccda02
us-west-1	ami-0c90cabe5dee3178
us-west-2	ami-0f96437a23aeae53f

6.13.14.1. AWS regions without a published RHCOS AMI

You can deploy an OpenShift Container Platform cluster to Amazon Web Services (AWS) regions without native support for a Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) or the AWS software development kit (SDK). If a published AMI is not available for an AWS region, you can upload a custom AMI prior to installing the cluster.

If you are deploying to a region not supported by the AWS SDK and you do not specify a custom AMI,

the installation program copies the **us-east-1** AMI to the user account automatically. Then the installation program creates the control plane machines with encrypted EBS volumes using the default or user-specified Key Management Service (KMS) key. This allows the AMI to follow the same process workflow as published RHCOS AMIs.

A region without native support for an RHCOS AMI is not available to select from the terminal during cluster creation because it is not published. However, you can install to this region by configuring the custom AMI in the **install-config.yaml** file.

6.13.14.2. Uploading a custom RHCOS AMI in AWS

If you are deploying to a custom Amazon Web Services (AWS) region, you must upload a custom Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) that belongs to that region.

Prerequisites

- You configured an AWS account.
- You created an Amazon S3 bucket with the required IAM [service role](#).
- You uploaded your RHCOS VMDK file to Amazon S3. The RHCOS VMDK file must be the highest version that is less than or equal to the OpenShift Container Platform version you are installing.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer](#).

Procedure

1. Export your AWS profile as an environment variable:

```
$ export AWS_PROFILE=<aws_profile> 1
```

2. Export the region to associate with your custom AMI as an environment variable:

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

3. Export the version of RHCOS you uploaded to Amazon S3 as an environment variable:

```
$ export RHCOS_VERSION=<version> 1
```

1 1 1 The RHCOS VMDK version, like **4.12.0**.

4. Export the Amazon S3 bucket name as an environment variable:

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. Create the **containers.json** file and define your RHCOS VMDK file:

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
```

```
"UserBucket": {
  "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
  "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
}
}
EOF
```

6. Import the RHCOS disk as an Amazon EBS snapshot:

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
  --description "<description>" \ 1
  --disk-container "file://<file_path>/containers.json" 2
```

- 1** The description of your RHCOS disk being imported, like **rhcos-\${RHCOS_VERSION}-x86_64-aws.x86_64**.
- 2** The file path to the JSON file describing your RHCOS disk. The JSON file should contain your Amazon S3 bucket name and key.

7. Check the status of the image import:

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

Example output

```
{
  "ImportSnapshotTasks": [
    {
      "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
      "ImportTaskId": "import-snap-fh6i8uil",
      "SnapshotTaskDetail": {
        "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
        "DiskImageSize": 819056640.0,
        "Format": "VMDK",
        "SnapshotId": "snap-06331325870076318",
        "Status": "completed",
        "UserBucket": {
          "S3Bucket": "external-images",
          "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
        }
      }
    }
  ]
}
```

Copy the **SnapshotId** to register the image.

8. Create a custom RHCOS AMI from the RHCOS snapshot:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
```

```

--ena-support \
--name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ ❸
--virtualization-type hvm \
--root-device-name '/dev/xvda' \
--block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' ❹

```

- ❶ The RHCOS VMDK architecture type, like **x86_64**, **aarch64**, **s390x**, or **ppc64le**.
- ❷ The **Description** from the imported snapshot.
- ❸ The name of the RHCOS AMI.
- ❹ The **SnapshotID** from the imported snapshot.

To learn more about these APIs, see the AWS documentation for [importing snapshots](#) and [creating EBS-backed AMIs](#).

6.13.15. Creating the bootstrap node in AWS

You must create the bootstrap node in Amazon Web Services (AWS) to use during OpenShift Container Platform cluster initialization. You do this by:

- Providing a location to serve the **bootstrap.ign** Ignition config file to your cluster. This file is located in your installation directory. The provided CloudFormation Template assumes that the Ignition config files for your cluster are served from an S3 bucket. If you choose to serve the files from another location, you must modify the templates.
- Using the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the bootstrap node that your OpenShift Container Platform installation requires.



NOTE

If you do not use the provided CloudFormation template to create your bootstrap node, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.

Procedure

1. Create the bucket by running the following command:

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

- 1** **<cluster-name>-infra** is the bucket name. When creating the **install-config.yaml** file, replace **<cluster-name>** with the name specified for the cluster.

You must use a presigned URL for your S3 bucket, instead of the **s3://** schema, if you are:

- Deploying to a region that has endpoints that differ from the AWS SDK.
- Deploying a proxy.
- Providing your own custom endpoints.

2. Upload the **bootstrap.ign** Ignition config file to the bucket by running the following command:

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

3. Verify that the file uploaded by running the following command:

```
$ aws s3 ls s3://<cluster-name>-infra/
```

Example output

```
2019-04-03 16:15:16 314878 bootstrap.ign
```



NOTE

The bootstrap Ignition config file does contain secrets, like X.509 keys. The following steps provide basic security for the S3 bucket. To provide additional security, you can enable an S3 bucket policy to allow only certain users, such as the OpenShift IAM user, to access objects that the bucket contains. You can avoid S3 entirely and serve your bootstrap Ignition config file from any address that the bootstrap machine can reach.

4. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcocAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
```

```

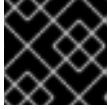
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupID", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcID", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbPTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbPTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 23
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]

```

- 1 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3 Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the bootstrap node based on your selected architecture.

- 4 Specify a valid **AWS::EC2::Image::Id** value.
 - 5 CIDR block to allow SSH access to the bootstrap node.
 - 6 Specify a CIDR block in the format **x.x.x.x/16-24**.
 - 7 The public subnet that is associated with your VPC to launch the bootstrap node into.
 - 8 Specify the **PublicSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 9 The master security group ID (for registering temporary rules)
 - 10 Specify the **MasterSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
 - 11 The VPC created resources will belong to.
 - 12 Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
 - 13 Location to fetch bootstrap Ignition config file from.
 - 14 Specify the S3 bucket and file name in the form **s3://<bucket_name>/bootstrap.ign**.
 - 15 Whether or not to register a network load balancer (NLB).
 - 16 Specify **yes** or **no**. If you specify **yes**, you must provide a Lambda Amazon Resource Name (ARN) value.
 - 17 The ARN for NLB IP target registration lambda group.
 - 18 Specify the **RegisterNlbTargetsLambda** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 19 The ARN for external API load balancer target group.
 - 20 Specify the **ExternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 21 The ARN for internal API load balancer target group.
 - 22 Specify the **InternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 23 The ARN for internal service load balancer target group.
 - 24 Specify the **InternalServiceTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
5. Copy the template from the **CloudFormation template for the bootstrap machines** section of this topic and save it as a YAML file on your computer. This template describes the bootstrap machine that your cluster requires.

6. Optional: If you are deploying the cluster with a proxy, you must update the ignition in the template to add the **ignition.config.proxy** fields. Additionally, if you have added the Amazon EC2, Elastic Load Balancing, and S3 VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
7. Launch the CloudFormation template to create a stack of AWS resources that represent the bootstrap node:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> ❶
  --template-body file://<template>.yaml ❷
  --parameters file://<parameters>.json ❸
  --capabilities CAPABILITY_NAMED_IAM ❹
```

- ❶ **<name>** is the name for the CloudFormation stack, such as **cluster-bootstrap**. You need the name of this stack if you remove the cluster.
- ❷ **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- ❸ **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.
- ❹ You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** and **AWS::IAM::InstanceProfile** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

8. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

Bootstrap InstanceId	The bootstrap Instance ID.
Bootstrap PublicIp	The bootstrap node public IP address.
Bootstrap PrivateIp	The bootstrap node private IP address.

6.13.15.1. CloudFormation template for the bootstrap machine

You can use the following CloudFormation template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster.

Example 6.47. CloudFormation template for the bootstrap machine

```

AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\_]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
    maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
    used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^((([0-9]{1-9}|[0-9]{2}|2[0-4][0-9]|25[0-5])\.)\.)\.)\.(3|([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
    4][0-9]|25[0-5])\V([0-9]{1-9}|2[0-9]|3[0-2]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
      - "yes"
      - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String

```

InternalApiTargetGroupArn:
 Description: ARN for internal API load balancer target group.
 Type: String

InternalServiceTargetGroupArn:
 Description: ARN for internal service load balancer target group.
 Type: String

BootstrapInstanceType:
 Description: Instance type for the bootstrap EC2 instance
 Default: "i3.large"
 Type: String

Metadata:

AWS::CloudFormation::Interface:

ParameterGroups:

- Label:

default: "Cluster Information"

Parameters:

- InfrastructureName

- Label:

default: "Host Information"

Parameters:

- RhcosAmi

- BootstrapIgnitionLocation

- MasterSecurityGroupId

- Label:

default: "Network Configuration"

Parameters:

- VpcId

- AllowedBootstrapSshCidr

- PublicSubnet

- Label:

default: "Load Balancer Automation"

Parameters:

- AutoRegisterELB

- RegisterNlbTargetsLambdaArn

- ExternalApiTargetGroupArn

- InternalApiTargetGroupArn

- InternalServiceTargetGroupArn

ParameterLabels:

InfrastructureName:

default: "Infrastructure Name"

VpcId:

default: "VPC ID"

AllowedBootstrapSshCidr:

default: "Allowed SSH Source"

PublicSubnet:

default: "Public Subnet"

RhcosAmi:

default: "Red Hat Enterprise Linux CoreOS AMI ID"

BootstrapIgnitionLocation:

default: "Bootstrap Ignition Source"

MasterSecurityGroupId:

default: "Master Security Group ID"

AutoRegisterELB:

default: "Use Provided ELB Automation"

Conditions:

DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:

BootstrapIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Path: "/"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action: "ec2:Describe*"

Resource: "*"

- Effect: "Allow"

Action: "ec2:AttachVolume"

Resource: "*"

- Effect: "Allow"

Action: "ec2:DetachVolume"

Resource: "*"

- Effect: "Allow"

Action: "s3:GetObject"

Resource: "*"

BootstrapInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Path: "/"

Roles:

- Ref: "BootstrapIamRole"

BootstrapSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Cluster Bootstrap Security Group

SecurityGroupIngress:

- IpProtocol: tcp

FromPort: 22

ToPort: 22

CidrIp: !Ref AllowedBootstrapSshCidr

- IpProtocol: tcp

ToPort: 19531

FromPort: 19531

CidrIp: 0.0.0.0/0

VpcId: !Ref VpcId

```

BootstrapInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    IamInstanceProfile: !Ref BootstrapInstanceProfile
    InstanceType: !Ref BootstrapInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "true"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "BootstrapSecurityGroup"
          - !Ref "MasterSecurityGroup"
        SubnetId: !Ref "PublicSubnet"
    UserData:
      Fn::Base64: !Sub
        - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
        - {
            S3Loc: !Ref BootstrapIgnitionLocation
          }

```

```

RegisterBootstrapApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

```

```

RegisterBootstrapInternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

```

```

RegisterBootstrapInternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

```

```

Outputs:
BootstrapInstanceid:
  Description: Bootstrap Instance ID.
  Value: !Ref BootstrapInstance

```

```

BootstrapPublicIp:
  Description: The bootstrap node public IP address.
  Value: !GetAtt BootstrapInstance.PublicIp

```

BootstrapPrivatelP:
 Description: The bootstrap node private IP address.
 Value: !GetAtt BootstrapInstance.PrivatelP

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).
- See [RHCOS AMIs for the AWS infrastructure](#) for details about the Red Hat Enterprise Linux CoreOS (RHCOS) AMIs for the AWS zones.

6.13.16. Creating the control plane machines in AWS

You must create the control plane machines in Amazon Web Services (AWS) that your cluster will use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the control plane nodes.



IMPORTANT

The CloudFormation template creates a stack that represents three control plane nodes.



NOTE

If you do not use the provided CloudFormation template to create your control plane nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.

Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
```

```

    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcoshAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
  {
    "ParameterKey": "Master0Subnet", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "Master1Subnet", 13
    "ParameterValue": "subnet-<random_string>" 14
  },
  {
    "ParameterKey": "Master2Subnet", 15
    "ParameterValue": "subnet-<random_string>" 16
  },
  {
    "ParameterKey": "MasterSecurityGroupID", 17
    "ParameterValue": "sg-<random_string>" 18
  },
  {
    "ParameterKey": "IgnitionLocation", 19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
  } 20
  {
    "ParameterKey": "CertificateAuthorities", 21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" 22
  },
  {
    "ParameterKey": "MasterInstanceProfileName", 23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
  },
  {
    "ParameterKey": "MasterInstanceType", 25
    "ParameterValue": "" 26
  },
  {
    "ParameterKey": "AutoRegisterELB", 27

```

```

    "ParameterValue": "yes" 28
  },
  {
    "ParameterKey": "RegisterNlbPTargetsLambdaArn", 29
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbPTargets-<random_string>" 30
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]

```

- 1 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3 Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the control plane machines based on your selected architecture.
- 4 Specify an **AWS::EC2::Image::Id** value.
- 5 Whether or not to perform DNS etcd registration.
- 6 Specify **yes** or **no**. If you specify **yes**, you must provide hosted zone information.
- 7 The Route 53 private zone ID to register the etcd targets with.
- 8 Specify the **PrivateHostedZoneId** value from the output of the CloudFormation template for DNS and load balancing.
- 9 The Route 53 zone to register the targets with.
- 10 Specify **<cluster_name>.<domain_name>** where **<domain_name>** is the Route 53 base domain that you used when you generated **install-config.yaml** file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
- 11 13 15 A subnet, preferably private, to launch the control plane machines on.
- 12 14 16 Specify a subnet from the **PrivateSubnets** value from the output of the CloudFormation template for DNS and load balancing.

- 17 The master security group ID to associate with control plane nodes.
- 18 Specify the **MasterSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
- 19 The location to fetch control plane Ignition config file from.
- 20 Specify the generated Ignition config file location, https://api-int.<cluster_name>.<domain_name>:22623/config/master.
- 21 The base64 encoded certificate authority string to use.
- 22 Specify the value from the **master.ign** file that is in the installation directory. This value is the long string with the format **data:text/plain;charset=utf-8;base64,ABC...xYz==**.
- 23 The IAM profile to associate with control plane nodes.
- 24 Specify the **MasterInstanceProfile** parameter value from the output of the CloudFormation template for the security group and roles.
- 25 The type of AWS instance to use for the control plane machines based on your selected architecture.
- 26 The instance type value corresponds to the minimum resource requirements for control plane machines. For example **m6i.xlarge** is a type for AMD64, and **m6g.xlarge** is a type for ARM64.
- 27 Whether or not to register a network load balancer (NLB).
- 28 Specify **yes** or **no**. If you specify **yes**, you must provide a Lambda Amazon Resource Name (ARN) value.
- 29 The ARN for NLB IP target registration lambda group.
- 30 Specify the **RegisterNlbTargetsLambda** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
- 31 The ARN for external API load balancer target group.
- 32 Specify the **ExternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
- 33 The ARN for internal API load balancer target group.
- 34 Specify the **InternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
- 35 The ARN for internal service load balancer target group.
- 36 Specify the **InternalServiceTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.

- Copy the template from the **CloudFormation template for control plane machines** section of this topic and save it as a YAML file on your computer. This template describes the control plane machines that your cluster requires.
- If you specified an **m5** instance type as the value for **MasterInstanceType**, add that instance type to the **MasterInstanceType.AllowedValues** parameter in the CloudFormation template.
- Launch the CloudFormation template to create a stack of AWS resources that represent the control plane nodes:

**IMPORTANT**

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
```

- 1** **<name>** is the name for the CloudFormation stack, such as **cluster-control-plane**. You need the name of this stack if you remove the cluster.
- 2** **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3** **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```

**NOTE**

The CloudFormation template creates a stack that represents three control plane nodes.

- Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6.13.16.1. CloudFormation template for control plane machines

You can use the following CloudFormation template to deploy the control plane machines that you need for your OpenShift Container Platform cluster.

Example 6.48. CloudFormation template for control plane machines

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)
```

Parameters:

InfrastructureName:

AllowedPattern: `^[a-zA-Z][a-zA-Z0-9\-\]{0,26}$`

MaxLength: `27`

MinLength: `1`

ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of `27` characters.

Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.

Type: String

RhcOSAmi:

Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.

Type: `AWS::EC2::Image::Id`

AutoRegisterDNS:

Default: `""`

Description: unused

Type: String

PrivateHostedZoneId:

Default: `""`

Description: unused

Type: String

PrivateHostedZoneName:

Default: `""`

Description: unused

Type: String

Master0Subnet:

Description: The subnets, recommend private, to launch the master nodes into.

Type: `AWS::EC2::Subnet::Id`

Master1Subnet:

Description: The subnets, recommend private, to launch the master nodes into.

Type: `AWS::EC2::Subnet::Id`

Master2Subnet:

Description: The subnets, recommend private, to launch the master nodes into.

Type: `AWS::EC2::Subnet::Id`

MasterSecurityGroupId:

Description: The master security group ID to associate with master nodes.

Type: `AWS::EC2::SecurityGroup::Id`

IgnitionLocation:

Default: `https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master`

Description: Ignition config file location.

Type: String

CertificateAuthorities:

Default: `data:text/plain;charset=utf-8;base64,ABC...xYz==`

Description: Base64 encoded certificate authority string to use.

Type: String

MasterInstanceProfileName:

Description: IAM profile to associate with master nodes.

Type: String

MasterInstanceType:

Default: `m5.xlarge`

Type: String

AutoRegisterELB:

Default: `"yes"`

AllowedValues:

- `"yes"`

- `"no"`

Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?

Type: String

RegisterNlbTargetsLambdaArn:

Description: ARN for NLB IP target registration lambda. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

ExternalApiTargetGroupArn:

Description: ARN for external API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

InternalApiTargetGroupArn:

Description: ARN for internal API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

InternalServiceTargetGroupArn:

Description: ARN for internal service load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

Metadata:

AWS::CloudFormation::Interface:

ParameterGroups:

- Label:

default: "Cluster Information"

Parameters:

- InfrastructureName

- Label:

default: "Host Information"

Parameters:

- MasterInstanceType

- RhcosAmi

- IgnitionLocation

- CertificateAuthorities

- MasterSecurityGroupId

- MasterInstanceProfileName

- Label:

default: "Network Configuration"

Parameters:

- VpcId

- AllowedBootstrapSshCidr

- Master0Subnet

- Master1Subnet

- Master2Subnet

- Label:

default: "Load Balancer Automation"

Parameters:

- AutoRegisterELB

- RegisterNlbTargetsLambdaArn

- ExternalApiTargetGroupArn

- InternalApiTargetGroupArn

- InternalServiceTargetGroupArn

ParameterLabels:

InfrastructureName:

default: "Infrastructure Name"

VpcId:

default: "VPC ID"

```

Master0Subnet:
  default: "Master-0 Subnet"
Master1Subnet:
  default: "Master-1 Subnet"
Master2Subnet:
  default: "Master-2 Subnet"
MasterInstanceType:
  default: "Master Instance Type"
MasterInstanceProfileName:
  default: "Master Instance Profile Name"
RhcosAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"
BootstrapIgnitionLocation:
  default: "Master Ignition Source"
CertificateAuthorities:
  default: "Ignition CA String"
MasterSecurityGroupId:
  default: "Master Security Group ID"
AutoRegisterELB:
  default: "Use Provided ELB Automation"

```

Conditions:

```
DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
```

Resources:

Master0:

```
Type: AWS::EC2::Instance
```

Properties:

```
ImageId: !Ref RhcosAmi
```

```
BlockDeviceMappings:
```

```
- DeviceName: /dev/xvda
```

```
  Ebs:
```

```
    VolumeSize: "120"
```

```
    VolumeType: "gp2"
```

```
IamInstanceProfile: !Ref MasterInstanceProfileName
```

```
InstanceType: !Ref MasterInstanceType
```

```
NetworkInterfaces:
```

```
- AssociatePublicIpAddress: "false"
```

```
  DeviceIndex: "0"
```

```
  GroupSet:
```

```
- !Ref "MasterSecurityGroupId"
```

```
  SubnetId: !Ref "Master0Subnet"
```

```
UserData:
```

```
Fn::Base64: !Sub
```

```
- '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}}, "version":"3.1.0"}'
```

```
- {
```

```
  SOURCE: !Ref IgnitionLocation,
```

```
  CA_BUNDLE: !Ref CertificateAuthorities,
```

```
}
```

```
Tags:
```

```
- Key: !Join [ "", ["kubernetes.io/cluster/", !Ref InfrastructureName] ]
```

```
  Value: "shared"
```

RegisterMaster0:

```
Condition: DoRegistration
```

Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref ExternalApiTargetGroupArn
 TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalApiTarget:
 Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref InternalApiTargetGroupArn
 TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalServiceTarget:
 Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref InternalServiceTargetGroupArn
 TargetIp: !GetAtt Master0.PrivateIp

Master1:
 Type: AWS::EC2::Instance
 Properties:
 ImageId: !Ref RhcosAmi
 BlockDeviceMappings:
 - DeviceName: /dev/xvda
 Ebs:
 VolumeSize: "120"
 VolumeType: "gp2"
 IamInstanceProfile: !Ref MasterInstanceProfileName
 InstanceType: !Ref MasterInstanceType
 NetworkInterfaces:
 - AssociatePublicIpAddress: "false"
 DeviceIndex: "0"
 GroupSet:
 - !Ref "MasterSecurityGroupId"
 SubnetId: !Ref "Master1Subnet"
 UserData:
 Fn::Base64: !Sub
 - '{"ignition":{"config":{"merge":{"source":"\${SOURCE}"}}, "security":{"tls":
 {"certificateAuthorities":[{"source":"\${CA_BUNDLE}"}]}}, "version":"3.1.0"}'
 - {
 SOURCE: !Ref IgnitionLocation,
 CA_BUNDLE: !Ref CertificateAuthorities,
 }
 Tags:
 - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
 Value: "shared"

RegisterMaster1:
 Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn

TargetArn: !Ref ExternalApiTargetGroupArn
 TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalApiTarget:

Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref InternalApiTargetGroupArn
 TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalServiceTarget:

Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref InternalServiceTargetGroupArn
 TargetIp: !GetAtt Master1.PrivateIp

Master2:

Type: AWS::EC2::Instance
 Properties:
 ImageId: !Ref RhcosAmi
 BlockDeviceMappings:
 - DeviceName: /dev/xvda
 Ebs:
 VolumeSize: "120"
 VolumeType: "gp2"
 IamInstanceProfile: !Ref MasterInstanceProfileName
 InstanceType: !Ref MasterInstanceType
 NetworkInterfaces:
 - AssociatePublicIpAddress: "false"
 DeviceIndex: "0"
 GroupSet:
 - !Ref "MasterSecurityGroupId"
 SubnetId: !Ref "Master2Subnet"
 UserData:
 Fn::Base64: !Sub
 - '{"ignition":{"config":{"merge":[{"source":"\${SOURCE}"}]},"security":{"tls":{"certificateAuthorities":[{"source":"\${CA_BUNDLE}"}]},"version":"3.1.0"}}'
 - {
 SOURCE: !Ref IgnitionLocation,
 CA_BUNDLE: !Ref CertificateAuthorities,
 }
 Tags:
 - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
 Value: "shared"

RegisterMaster2:

Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref ExternalApiTargetGroupArn
 TargetIp: !GetAtt Master2.PrivateIp

```

RegisterMaster2InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

```

```

RegisterMaster2InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

```

Outputs:

PrivateIPs:

Description: The control-plane node private IP addresses.

Value:

```

!Join [
  ",",
  [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
]

```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).

6.13.17. Creating the worker nodes in AWS

You can create worker nodes in Amazon Web Services (AWS) for your cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent a worker node.



IMPORTANT

The CloudFormation template creates a stack that represents one worker node. You must create a stack for each worker node.



NOTE

If you do not use the provided CloudFormation template to create your worker nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.

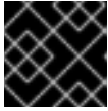
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.

Procedure

1. Create a JSON file that contains the parameter values that the CloudFormation template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcOsAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupID", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
    10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
    "ParameterValue": "" 16
  }
]
```

- 1 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
 - 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
 - 3 Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the worker nodes based on your selected architecture.
 - 4 Specify an **AWS::EC2::Image::Id** value.
 - 5 A subnet, preferably private, to start the worker nodes on.
 - 6 Specify a subnet from the **PrivateSubnets** value from the output of the CloudFormation template for DNS and load balancing.
 - 7 The worker security group ID to associate with worker nodes.
 - 8 Specify the **WorkerSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
 - 9 The location to fetch the bootstrap Ignition config file from.
 - 10 Specify the generated Ignition config location, https://api-int.<cluster_name>.<domain_name>:22623/config/worker.
 - 11 Base64 encoded certificate authority string to use.
 - 12 Specify the value from the **worker.ign** file that is in the installation directory. This value is the long string with the format **data:text/plain;charset=utf-8;base64,ABC...xYz==**.
 - 13 The IAM profile to associate with worker nodes.
 - 14 Specify the **WorkerInstanceProfile** parameter value from the output of the CloudFormation template for the security group and roles.
 - 15 The type of AWS instance to use for the compute machines based on your selected architecture.
 - 16 The instance type value corresponds to the minimum resource requirements for compute machines. For example **m6i.large** is a type for AMD64. and **m6g.large** is a type for ARM64.
2. Copy the template from the **CloudFormation template for worker machines** section of this topic and save it as a YAML file on your computer. This template describes the networking objects and load balancers that your cluster requires.
 3. Optional: If you specified an **m5** instance type as the value for **WorkerInstanceType**, add that instance type to the **WorkerInstanceType.AllowedValues** parameter in the CloudFormation template.
 4. Optional: If you are deploying with an AWS Marketplace image, update the **Worker0.type.properties.ImageId** parameter with the AMI ID that you obtained from your subscription.
 5. Use the CloudFormation template to create a stack of AWS resources that represent a worker node:

**IMPORTANT**

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml \ 2
  --parameters file://<parameters>.json 3
```

- 1** **<name>** is the name for the CloudFormation stack, such as **cluster-worker-1**. You need the name of this stack if you remove the cluster.
- 2** **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3** **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-11eb-348f-sd9888c65b59
```

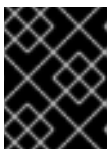
**NOTE**

The CloudFormation template creates a stack that represents one worker node.

6. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

7. Continue to create worker stacks until you have created enough worker machines for your cluster. You can create additional worker stacks by referencing the same template and parameter files and specifying a different stack name.

**IMPORTANT**

You must create at least two worker machines, so you must create at least two stacks that use this CloudFormation template.

6.13.17.1. CloudFormation template for worker machines

You can use the following CloudFormation template to deploy the worker machines that you need for your OpenShift Container Platform cluster.

Example 6.49. CloudFormation template for worker machines

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
```

AllowedPattern: `^[a-zA-Z][a-zA-Z0-9-]{0,26}$`

MaxLength: `27`

MinLength: `1`

ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of `27` characters.

Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.

Type: String

RhcosAmi:

Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.

Type: `AWS::EC2::Image::Id`

Subnet:

Description: The subnets, recommend private, to launch the master nodes into.

Type: `AWS::EC2::Subnet::Id`

WorkerSecurityGroupId:

Description: The master security group ID to associate with master nodes.

Type: `AWS::EC2::SecurityGroup::Id`

IgnitionLocation:

Default: `https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker`

Description: Ignition config file location.

Type: String

CertificateAuthorities:

Default: `data:text/plain;charset=utf-8;base64,ABC...xYz==`

Description: Base64 encoded certificate authority string to use.

Type: String

WorkerInstanceProfileName:

Description: IAM profile to associate with master nodes.

Type: String

WorkerInstanceType:

Default: `m5.large`

Type: String

Metadata:

`AWS::CloudFormation::Interface`:

ParameterGroups:

- Label:

default: `"Cluster Information"`

Parameters:

- `InfrastructureName`

- Label:

default: `"Host Information"`

Parameters:

- `WorkerInstanceType`

- `RhcosAmi`

- `IgnitionLocation`

- `CertificateAuthorities`

- `WorkerSecurityGroupId`

- `WorkerInstanceProfileName`

- Label:

default: `"Network Configuration"`

Parameters:

- `Subnet`

ParameterLabels:

Subnet:

default: `"Subnet"`

InfrastructureName:

default: `"Infrastructure Name"`

```

WorkerInstanceType:
  default: "Worker Instance Type"
WorkerInstanceProfileName:
  default: "Worker Instance Profile Name"
RhcocAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"
IgnitionLocation:
  default: "Worker Ignition Source"
CertificateAuthorities:
  default: "Ignition CA String"
WorkerSecurityGroupId:
  default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcocAmi
      BlockDeviceMappings:
        - DeviceName: /dev/xvda
          Ebs:
            VolumeSize: "120"
            VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
        - AssociatePublicIp: "false"
          DeviceIndex: "0"
          GroupSet:
            - !Ref "WorkerSecurityGroupId"
          SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
        - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
          Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp

```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).

6.13.18. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure

After you create all of the required infrastructure in Amazon Web Services (AWS), you can start the bootstrap sequence that initializes the OpenShift Container Platform control plane.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.
- You created the worker nodes.

Procedure

1. Change to the directory that contains the installation program and start the bootstrap process that initializes the OpenShift Container Platform control plane:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1  
--log-level=info 2
```

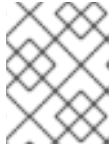
1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 20m0s for the Kubernetes API at  
https://api.mycluster.example.com:6443...  
INFO API v1.25.0 up  
INFO Waiting up to 30m0s for bootstrapping to complete...  
INFO It is now safe to remove the bootstrap resources  
INFO Time elapsed: 1s
```

If the command exits without a **FATAL** warning, your OpenShift Container Platform control plane has initialized.

**NOTE**

After the control plane initializes, it sets up the compute nodes and installs additional services in the form of Operators.

Additional resources

- See [Monitoring installation progress](#) for details about monitoring the installation, bootstrap, and control plane logs as an OpenShift Container Platform installation progresses.
- See [Gathering bootstrap node diagnostic data](#) for information about troubleshooting issues related to the bootstrap process.
- You can view details about the running instances that are created by using the [AWS EC2 console](#).

6.13.19. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

■

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

6.13.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.13.21. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
```

```

master-0 Ready   master 63m v1.25.0
master-1 Ready   master 63m v1.25.0
master-2 Ready   master 64m v1.25.0

```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```

NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br 15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...

```

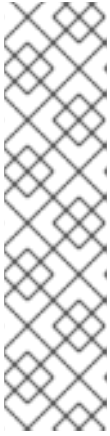
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

6.13.22. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

- Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m

console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

6.13.22.1. Image registry storage configuration

Amazon Web Services provides default storage, which means the Image Registry Operator is available after installation. However, if the Registry Operator cannot create an S3 bucket and automatically configure storage, you must manually configure registry storage.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

You can configure registry storage for user-provisioned infrastructure in AWS to deploy OpenShift Container Platform to hidden regions. See [Configuring the registry for AWS user-provisioned infrastructure](#) for more information.

6.13.22.1.1. Configuring registry storage for AWS with user-provisioned infrastructure

During installation, your cloud credentials are sufficient to create an Amazon S3 bucket and the Registry Operator will automatically configure storage.

If the Registry Operator cannot create an S3 bucket and automatically configure storage, you can create an S3 bucket and configure storage with the following procedure.

Prerequisites

- You have a cluster on AWS with user-provisioned infrastructure.
- For Amazon S3 storage, the secret is expected to contain two keys:
 - **REGISTRY_STORAGE_S3_ACCESSKEY**
 - **REGISTRY_STORAGE_S3_SECRETKEY**

Procedure

Use the following procedure if the Registry Operator cannot create an S3 bucket and automatically configure storage.

1. Set up a [Bucket Lifecycle Policy](#) to abort incomplete multipart uploads that are one day old.
2. Fill in the storage configuration in **configs.imageregistry.operator.openshift.io/cluster**:

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

Example configuration

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```



WARNING

To secure your registry images in AWS, [block public access](#) to the S3 bucket.

6.13.22.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

6.13.23. Deleting the bootstrap resources

After you complete the initial Operator configuration for the cluster, remove the bootstrap resources from Amazon Web Services (AWS).

Prerequisites

- You completed the initial Operator configuration for your cluster.

Procedure

- Delete the bootstrap resources. If you used the CloudFormation template, [delete its stack](#):

- Delete the stack by using the AWS CLI:

```
$ aws cloudformation delete-stack --stack-name <name> 1
```

1 **<name>** is the name of your bootstrap stack.

- Delete the stack by using the [AWS CloudFormation console](#).

6.13.24. Creating the Ingress DNS Records

If you removed the DNS Zone configuration, manually create DNS records that point to the Ingress load balancer. You can create either a wildcard record or specific records. While the following procedure uses A records, you can use other record types that you require, such as CNAME or alias.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Amazon Web Services (AWS) that uses infrastructure that you provisioned.
- You installed the OpenShift CLI (**oc**).
- You installed the **jq** package.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#).

Procedure

- Determine the routes to create.
 - To create a wildcard record, use ***.apps.<cluster_name>.<domain_name>**, where **<cluster_name>** is your cluster name, and **<domain_name>** is the Route 53 base domain for your OpenShift Container Platform cluster.

- To create specific records, you must create a record for each route that your cluster uses, as shown in the output of the following command:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

- Retrieve the Ingress Operator load balancer status and note the value of the external IP address that it uses, which is shown in the **EXTERNAL-IP** column:

```
$ oc -n openshift-ingress get service router-default
```

Example output

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
router-default	LoadBalancer	172.30.62.215	ab3...28.us-east-2.elb.amazonaws.com	80:31499/TCP,443:30693/TCP
		5m		

- Locate the hosted zone ID for the load balancer:

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
"<external_ip>").CanonicalHostedZoneNameID' 1
```

- For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer that you obtained.

Example output

```
Z3AADJGX6KTTL2
```

The output of this command is the load balancer hosted zone ID.

- Obtain the public hosted zone ID for your cluster's domain:

```
$ aws route53 list-hosted-zones-by-name \
  --dns-name "<domain_name>" 1 \
  --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2 \
  --output text
```

- For **<domain_name>**, specify the Route 53 base domain for your OpenShift Container Platform cluster.

Example output

```
/hostedzone/Z3URY6TWQ91KVV
```

The public hosted zone ID for your domain is shown in the command output. In this example, it is **Z3URY6TWQ91KVV**.

5. Add the alias records to your private zone:

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ 1
> "Changes": [
> {
>   "Action": "CREATE",
>   "ResourceRecordSet": {
>     "Name": "\\052.apps.<cluster_domain>", 2
>     "Type": "A",
>     "AliasTarget":{
>       "HostedZoneId": "<hosted_zone_id>", 3
>       "DNSName": "<external_ip>.", 4
>       "EvaluateTargetHealth": false
>     }
>   }
> }
> ]
> }'
```

- 1 For **<private_hosted_zone_id>**, specify the value from the output of the CloudFormation template for DNS and load balancing.
- 2 For **<cluster_domain>**, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.
- 3 For **<hosted_zone_id>**, specify the public hosted zone ID for the load balancer that you obtained.
- 4 For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6. Add the records to your public zone:

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --
change-batch '{ 1
> "Changes": [
> {
>   "Action": "CREATE",
>   "ResourceRecordSet": {
>     "Name": "\\052.apps.<cluster_domain>", 2
>     "Type": "A",
>     "AliasTarget":{
>       "HostedZoneId": "<hosted_zone_id>", 3
>       "DNSName": "<external_ip>.", 4
>       "EvaluateTargetHealth": false
```

```
>     }
>   }
> }
> ]
> }'
```

- 1 For **<public_hosted_zone_id>**, specify the public hosted zone for your domain.
- 2 For **<cluster_domain>**, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.
- 3 For **<hosted_zone_id>**, specify the public hosted zone ID for the load balancer that you obtained.
- 4 For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6.13.25. Completing an AWS installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Amazon Web Service (AWS) user-provisioned infrastructure, monitor the deployment to completion.

Prerequisites

- You removed the bootstrap node for an OpenShift Container Platform cluster on user-provisioned AWS infrastructure.
- You installed the **oc** CLI.

Procedure

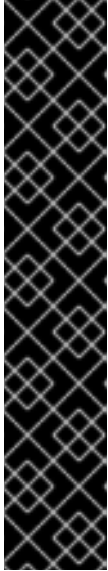
- From the directory that contains the installation program, complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 1s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.13.26. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console
https reencrypt/Redirect None
```

-
- 3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.13.27. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

6.13.28. Additional resources

- See [Working with stacks](#) in the AWS documentation for more information about AWS CloudFormation stacks.

6.13.29. Next steps

- [Validating an installation](#).
- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

6.14. INSTALLING A CLUSTER USING AWS LOCAL ZONES

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) into an existing VPC, extending workers to the edge of the Cloud Infrastructure using AWS Local Zones.

After you create an Amazon Web Service (AWS) Local Zone environment, and you deploy your cluster, you can use edge worker nodes to create user workloads in Local Zone subnets.

AWS Local Zones are a type of infrastructure that place Cloud Resources close to the metropolitan regions. For more information, see the [AWS Local Zones Documentation](#).

OpenShift Container Platform can be installed in existing VPCs with Local Zone subnets. The Local Zone subnets can be used to extend the regular workers' nodes to the edge networks. The edge worker nodes are dedicated to running user workloads.

One way to create the VPC and subnets is to use the provided CloudFormation templates. You can modify the templates to customize your infrastructure or use the information that they contain to create AWS objects according to your company's policies.



IMPORTANT

The steps for performing an installer-provisioned infrastructure installation are provided as an example only. Installing a cluster with VPC you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. The CloudFormation templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

6.14.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- You noted the region and supported [AWS Local Zones locations](#) to create the network resources in.
- You read the [Features](#) for each AWS Local Zones location.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or UNIX\)](#) in the AWS documentation.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.14.2. Cluster limitations in AWS Local Zones

Some limitations exist when you attempt to deploy a cluster with a default installation configuration in Amazon Web Services (AWS) Local Zones.



IMPORTANT

The following list details limitations when deploying a cluster in AWS Local Zones:

- The Maximum Transmission Unit (MTU) between an Amazon EC2 instance in a Local Zone and an Amazon EC2 instance in the Region is **1300**. This causes the cluster-wide network MTU to change according to the network plugin that is used on the deployment.
- Network resources such as Network Load Balancer (NLB), Classic Load Balancer, and Network Address Translation (NAT) Gateways are not supported in AWS Local Zones.
- For an OpenShift Container Platform cluster on AWS, the AWS Elastic Block Storage (EBS) **gp3** type volume is the default for node volumes and the default for the storage class. This volume type is not globally available on Local Zone locations. By default, the nodes running in Local Zones are deployed with the **gp2** EBS volume. The **gp2-csi StorageClass** must be set when creating workloads on Local Zone nodes.

Additional resources

- [Storage classes](#)

6.14.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.14.4. Opting into AWS Local Zones

If you plan to create the subnets in AWS Local Zones, you must opt in to each zone group separately.

Prerequisites

- You have installed the AWS CLI.
- You have determined into which region you will deploy your OpenShift Container Platform cluster.

Procedure

1. Export a variable to contain the name of the region in which you plan to deploy your OpenShift Container Platform cluster by running the following command:

```
$ export CLUSTER_REGION="<region_name>" 1
```

- 1** For **<region_name>**, specify a valid AWS region name, such as **us-east-1**.

2. List the zones that are available in your region by running the following command:

```
$ aws --region ${CLUSTER_REGION} ec2 describe-availability-zones \
  --query 'AvailabilityZones[.][{ZoneName: ZoneName, GroupName: GroupName, Status: OptInStatus}]' \
  --filters Name=zone-type,Values=local-zone \
  --all-availability-zones
```

Depending on the region, the list of available zones can be long. The command will return the following fields:

ZoneName

The name of the Local Zone.

GroupName

The group that the zone is part of. You need to save this name to opt in.

Status

The status of the Local Zone group. If the status is **not-opted-in**, you must opt in the **GroupName** by running the commands that follow.

3. Export a variable to contain the name of the Local Zone to host your VPC by running the following command:

```
$ export ZONE_GROUP_NAME="<value_of_GroupName>" 1
```

- 1** The **<value_of_GroupName>** specifies the name of the group of the Local Zone you want to create subnets on. For example, specify **us-east-1-nyc-1** to use the zone **us-east-1-nyc-1a**, US East (New York).

4. Opt in to the zone group on your AWS account by running the following command:

```
$ aws ec2 modify-availability-zone-group \
  --group-name "${ZONE_GROUP_NAME}" \
  --opt-in-status opted-in
```

6.14.5. Obtaining an AWS Marketplace image

If you are deploying an OpenShift Container Platform cluster using an AWS Marketplace image, you must first subscribe through AWS. Subscribing to the offer provides you with the AMI ID that the installation program uses to deploy worker nodes.

Prerequisites

- You have an AWS account to purchase the offer. This account does not have to be the same account that is used to install the cluster.

Procedure

1. Complete the OpenShift Container Platform subscription from the [AWS Marketplace](#).
2. Record the AMI ID for your specific region. As part of the installation process, you must update the **install-config.yaml** file with this value before deploying the cluster.

Sample `install-config.yaml` file with AWS Marketplace worker nodes

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 1
      type: m5.4xlarge
    replicas: 3
  metadata:
    name: test-cluster
  platform:
    aws:
      region: us-east-2 2
  sshKey: ssh-ed25519 AAAA...
  pullSecret: '{"auths": ...}'
```

- 1** The AMI ID from your AWS Marketplace subscription.
- 2** Your AMI ID is associated with a specific AWS region. When creating the installation configuration file, ensure that you select the same AWS region that you specified when configuring your subscription.

6.14.6. Creating a VPC that uses AWS Local Zones

You must create a Virtual Private Cloud (VPC), and subnets for each Local Zone location, in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to extend worker nodes to the edge locations. You can further customize the VPC to meet your requirements, including VPN, route tables, and add new Local Zone subnets that are not included at initial deployment.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the VPC.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You opted in to the AWS Local Zones on your AWS account.

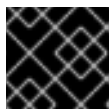
Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 5
    "ParameterValue": "3" 6
  },
  {
    "ParameterKey": "SubnetBits", 7
    "ParameterValue": "12" 8
  }
]
```

- 1** A short, representative cluster name to use for hostnames, etc.
- 2** Specify the cluster name that you used when you generated the **install-config.yaml** file for the cluster.
- 3** The CIDR block for the VPC.
- 4** Specify a CIDR block in the format **x.x.x.x/16-24**.
- 5** The number of availability zones to deploy the VPC in.
- 6** Specify an integer between **1** and **3**.
- 7** The size of each subnet in each availability zone.
- 8** Specify an integer between **5** and **13**, where **5** is **/27** and **13** is **/19**.

- Copy the template from the **CloudFormation template for the VPC** section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.
- Launch the CloudFormation template to create a stack of AWS resources that represent the VPC by running the following command:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> \ 1
  --template-body file://<template>.yaml \ 2
  --parameters file://<parameters>.json 3
```

- <name>** is the name for the CloudFormation stack, such as **cluster-vpc**. You need the name of this stack if you remove the cluster.
- <template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- <parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

- Confirm that the template components exist by running the following command:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

VpcId	The ID of your VPC.
PublicSubnetIds	The IDs of the new public subnets.
PrivateSubnetIds	The IDs of the new private subnets.
PublicRouteTableId	The ID of the new public route table ID.

6.14.6.1. CloudFormation template for the VPC that uses AWS Local Zones

You can use the following CloudFormation template to deploy the VPC that you need for your OpenShift Container Platform cluster that uses AWS Local Zones.

Example 6.50. CloudFormation template for the VPC

```

AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  ClusterName:
    Type: String
    Description: ClusterName used to prefix resource names
  VpcCidr:
    AllowedPattern: ^((([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.)\.)\{3\}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\)(\v(1[6-9]|2[0-4]))\)$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Network Configuration"
        Parameters:
          - VpcCidr
          - SubnetBits
      - Label:
          default: "Availability Zones"
        Parameters:
          - AvailabilityZoneCount
    ParameterLabels:
      ClusterName:
        default: ""
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

```

Conditions:

DoAz3: !Equals [3, !Ref AvailabilityZoneCount]

DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:

VPC:

Type: "AWS::EC2::VPC"

Properties:

EnableDnsSupport: "true"

EnableDnsHostnames: "true"

CidrBlock: !Ref VpcCidr

Tags:

- Key: Name

Value: !Join ["", [!Ref ClusterName, "-vpc"]]

- Key: !Join ["", ["kubernetes.io/cluster/unmanaged"]]

Value: "shared"

PublicSubnet:

Type: "AWS::EC2::Subnet"

Properties:

Vpclid: !Ref VPC

CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]

AvailabilityZone: !Select

- 0

- Fn::GetAZs: !Ref "AWS::Region"

Tags:

- Key: Name

Value: !Join ["", [!Ref ClusterName, "-public-1"]]

PublicSubnet2:

Type: "AWS::EC2::Subnet"

Condition: DoAz2

Properties:

Vpclid: !Ref VPC

CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]

AvailabilityZone: !Select

- 1

- Fn::GetAZs: !Ref "AWS::Region"

Tags:

- Key: Name

Value: !Join ["", [!Ref ClusterName, "-public-2"]]

PublicSubnet3:

Type: "AWS::EC2::Subnet"

Condition: DoAz3

Properties:

Vpclid: !Ref VPC

CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]

AvailabilityZone: !Select

- 2

- Fn::GetAZs: !Ref "AWS::Region"

Tags:

- Key: Name

Value: !Join ["", [!Ref ClusterName, "-public-3"]]

InternetGateway:

Type: "AWS::EC2::InternetGateway"

```

Properties:
  Tags:
  - Key: Name
    Value: !Join [ "", [ !Ref ClusterName, "-igw" ] ]
GatewayToInternet:
  Type: "AWS::EC2::VPCGatewayAttachment"
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref InternetGateway

PublicRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
    Tags:
    - Key: Name
      Value: !Join [ "", [ !Ref ClusterName, "-rtb-public" ] ]
PublicRoute:
  Type: "AWS::EC2::Route"
  DependsOn: GatewayToInternet
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet2
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable

PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 0
    - Fn::GetAZs: !Ref "AWS::Region"
    Tags:
    - Key: Name
      Value: !Join [ "", [ !Ref ClusterName, "-private-1" ] ]
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
    Tags:

```

```

- Key: Name
  Value: !Join [ "", [ !Ref ClusterName, "-rtb-private-1" ] ]
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
    Tags:
      - Key: Name
        Value: !Join [ "", [ !Ref ClusterName, "-natgw-private-1" ] ]
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT

PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
    Tags:
      - Key: Name
        Value: !Join [ "", [ !Ref ClusterName, "-private-2" ] ]
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Join [ "", [ !Ref ClusterName, "-rtb-private-2" ] ]
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2

```

```

Properties:
  SubnetId: !Ref PrivateSubnet2
  RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
  Tags:
  - Key: Name
    Value: !Join [ "", [ !Ref ClusterName, "-natgw-private-2" ] ]
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
  Tags:
  - Key: Name
    Value: !Join [ "", [ !Ref ClusterName, "-eip-private-2" ] ]
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2

PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
  Tags:
  - Key: Name
    Value: !Join [ "", [ !Ref ClusterName, "-private-3" ] ]
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
  Tags:
  - Key: Name
    Value: !Join [ "", [ !Ref ClusterName, "-rtb-private-3" ] ]
PrivateSubnetRouteTableAssociation3:

```

```

Type: "AWS::EC2::SubnetRouteTableAssociation"
Condition: DoAz3
Properties:
  SubnetId: !Ref PrivateSubnet3
  RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
    Tags:
      - Key: Name
        Value: !Join [ "", [ !Ref ClusterName, "-natgw-private-3" ] ]
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
    Tags:
      - Key: Name
        Value: !Join [ "", [ !Ref ClusterName, "-eip-private-3" ] ]
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
      - Effect: Allow
        Principal: '*'
        Action:
          - '*'
        Resource:
          - '*'
    RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
    ServiceName: !Join
      - "

```



```

- - com.amazonaws.
- !Ref 'AWS::Region'
- .s3
VpcId: !Ref VPC

```

Outputs:

VpcId:

Description: ID of the new VPC.

Value: !Ref VPC

PublicSubnetIds:

Description: Subnet IDs of the public subnets.

Value:

```
!Join [
```

```
  ",",
```

```
  [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]

```

```
]
```

PrivateSubnetIds:

Description: Subnet IDs of the private subnets.

Value:

```
!Join [
```

```
  ",",
```

```
  [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]

```

```
]
```

PublicRouteTableId:

Description: Public Route table ID

Value: !Ref PublicRouteTable

PrivateRouteTableId:

Description: Private Route table ID

Value: !Ref PrivateRouteTable

6.14.7. Creating a subnet in AWS Local Zones

You must create a subnet in AWS Local Zones before you configure a worker machineset for your OpenShift Container Platform cluster.

You must repeat the following process for each Local Zone you want to deploy worker nodes to.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the subnet.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.

- You opted in to the Local Zone group.

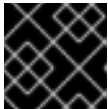
Procedure

- Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "VpcId", 3
    "ParameterValue": "vpc-<random_string>" 4
  },
  {
    "ParameterKey": "PublicRouteTableId", 5
    "ParameterValue": "<vpc_rtb_pub>" 6
  },
  {
    "ParameterKey": "LocalZoneName", 7
    "ParameterValue": "<cluster_region_name>-<location_identifier>-<zone_identifier>" 8
  },
  {
    "ParameterKey": "LocalZoneNameShort", 9
    "ParameterValue": "<lz_zone_shortname>" 10
  },
  {
    "ParameterKey": "PublicSubnetCidr", 11
    "ParameterValue": "10.0.128.0/20" 12
  }
]
```

- A short, representative cluster name to use for hostnames, etc.
- Specify the cluster name that you used when you generated the **install-config.yaml** file for the cluster.
- The VPC ID in which the Local Zone's subnet will be created.
- Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
- The Public Route Table ID for the VPC.
- Specify the **PublicRouteTableId** value from the output of the CloudFormation template for the VPC.
- The Local Zone name that the VPC belongs to.
- Specify the Local Zone that you opted your AWS account into, such as **us-east-1-nyc-1a**.
- The shortname of the AWS Local Zone that the VPC belongs to.
- Specify a short name for the AWS Local Zone that you opted your AWS account into, such

- 11 The CIDR block to allow access to the Local Zone.
 - 12 Specify a CIDR block in the format **x.x.x.x/16-24**.
2. Copy the template from the **CloudFormation template for the subnet** section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.
 3. Launch the CloudFormation template to create a stack of AWS resources that represent the VPC by running the following command:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <subnet_stack_name> \ 1
  --template-body file://<template>.yaml \ 2
  --parameters file://<parameters>.json 3
```

- 1 **<subnet_stack_name>** is the name for the CloudFormation stack, such as **cluster-lz-local_zone_shortcode**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-lz-nyc1/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. Confirm that the template components exist by running the following command:

```
$ aws cloudformation describe-stacks --stack-name <subnet_stack_name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

PublicSubnetIds	The IDs of the new public subnets.
------------------------	------------------------------------

6.14.7.1. CloudFormation template for the subnet that uses AWS Local Zones

You can use the following CloudFormation template to deploy the subnet that you need for your OpenShift Container Platform cluster that uses AWS Local Zones.

Example 6.51. CloudFormation template for the subnet

CloudFormation template used to create Local Zone subnets and dependencies

AWSTemplateFormatVersion: 2010-09-09

Description: Template for Best Practice VPC with 1-3 AZs

Parameters:

ClusterName:

Description: ClusterName used to prefix resource names

Type: String

VpcId:

Description: VPC Id

Type: String

LocalZoneName:

Description: Local Zone Name (Example us-east-1-bos-1)

Type: String

LocalZoneNameShort:

Description: Short name for Local Zone used on tag Name (Example bos1)

Type: String

PublicRouteTableId:

Description: Public Route Table ID to associate the Local Zone subnet

Type: String

PublicSubnetCidr:

AllowedPattern: ^((([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.)\.)\{3\}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\)(\|(1[6-9]|2[0-4]))\)\$

ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.

Default: 10.0.128.0/20

Description: CIDR block for Public Subnet

Type: String

Resources:

PublicSubnet:

Type: "AWS::EC2::Subnet"

Properties:

VpcId: !Ref VpcId

CidrBlock: !Ref PublicSubnetCidr

AvailabilityZone: !Ref LocalZoneName

Tags:

- Key: Name

Value: !Join

- ""

- [!Ref ClusterName, "-public-", !Ref LocalZoneNameShort, "-1"]

- Key: kubernetes.io/cluster/unmanaged

Value: "true"

PublicSubnetRouteTableAssociation:

Type: "AWS::EC2::SubnetRouteTableAssociation"

Properties:

SubnetId: !Ref PublicSubnet

RouteTableId: !Ref PublicRouteTableId

Outputs:

PublicSubnetIds:

Description: Subnet IDs of the public subnets.

Value:

!Join [

```

    ""',
    [!Ref PublicSubnet]
  ]

```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the [AWS CloudFormation console](#).

6.14.8. Obtaining the installation program

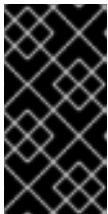
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

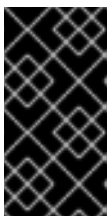
Procedure

- Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
- Select your infrastructure provider.
- Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

- Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.14.9. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.14.10. Creating the installation files for AWS

To install OpenShift Container Platform on Amazon Web Services (AWS) and use AWS Local Zones, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file and Kubernetes manifests.

6.14.10.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.50. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.14.10.2. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform for use with AWS Local Zones.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.52. Machine types based on 64-bit x86 architecture for AWS Local Zones

- **c5.***
- **c5d.***

- **m6i.***
- **m5.***
- **r5.***
- **t3.***

Additional resources

- See [AWS Local Zones features](#) in the AWS documentation for more information about AWS Local Zones and the supported instances types and services.

6.14.10.3. Creating the installation configuration file

Generate and customize the installation configuration file that the installation program needs to deploy your cluster.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You checked that you are deploying your cluster to a region with an accompanying Red Hat Enterprise Linux CoreOS (RHCOS) AMI published by Red Hat. If you are deploying to a region that requires a custom AMI, such as an AWS GovCloud region, you must create the **install-config.yaml** file manually.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.



IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

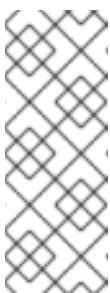
- b. At the prompts, provide the configuration details for your cloud:

- i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **aws** as the platform to target.
- iii. If you do not have an AWS profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

**NOTE**

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

- iv. Select the AWS region to deploy the cluster to. The region that you specify must be the same region that contains the Local Zone that you opted into for your AWS account.
 - v. Select the base domain for the Route 53 service that you configured for your cluster.
 - vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Edit the **install-config.yaml** file to provide the subnets for the availability zones that your VPC uses:

```
platform:
  aws:
    subnets: 1
    - publicSubnetId-1
    - publicSubnetId-2
    - publicSubnetId-3
    - privateSubnetId-1
    - privateSubnetId-2
    - privateSubnetId-3
```

- 1 Add the **subnets** section and specify the **PrivateSubnetIds** and **PublicSubnetIds** values from the outputs of the CloudFormation template for the VPC. Do not include the Local Zone subnets here.

- 3. Optional: Back up the **install-config.yaml** file.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- See [Configuration and credential file settings](#) in the AWS documentation for more information about AWS profile and credential configuration.

6.14.10.4. Creating the Kubernetes manifest files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest files that the cluster needs to configure the machines.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.
- You installed the **jq** package.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster by running the following command:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Set the default Maximum Transmission Unit (MTU) according to the network plugin:



IMPORTANT

Generally, the Maximum Transmission Unit (MTU) between an Amazon EC2 instance in a Local Zone and an Amazon EC2 instance in the Region is 1300. See [How Local Zones work](#) in the AWS documentation. The cluster network MTU must be always less than the EC2 MTU to account for the overhead. The specific overhead is determined by your network plugin, for example:

- OVN-Kubernetes: **100 bytes**
- OpenShift SDN: **50 bytes**

The network plugin could provide additional features, like IPsec, that also must be decreased the MTU. Check the documentation for additional information.

- a. If you are using the **OVN-Kubernetes** network plugin, enter the following command:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
```

```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      mtu: 1200
EOF

```

- b. If you are using the **OpenShift SDN** network plugin, enter the following command:

```

$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      mtu: 1250
EOF

```

3. Create the machine set manifests for the worker nodes in your Local Zone.

- a. Export a local variable that contains the name of the Local Zone that you opted your AWS account into by running the following command:

```
$ export LZ_ZONE_NAME="<local_zone_name>" 1
```

- 1** For **<local_zone_name>**, specify the Local Zone that you opted your AWS account into, such as **us-east-1-nyc-1a**.

- b. Review the instance types for the location that you will deploy to by running the following command:

```

$ aws ec2 describe-instance-type-offerings \
  --location-type availability-zone \
  --filters Name=location,Values=${LZ_ZONE_NAME} \
  --region <region> 1

```

- 1** For **<region>**, specify the name of the region that you will deploy to, such as **us-east-1**.

- c. Export a variable to define the instance type for the worker machines to deploy on the Local Zone subnet by running the following command:

```
$ export INSTANCE_TYPE="<instance_type>" 1
```

- 1** Set **<instance_type>** to a tested instance type, such as **c5d.2xlarge**.

- d. Store the AMI ID as a local variable by running the following command:

```
$ export AMI_ID=$(grep ami
<installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-0.yaml \
| tail -n1 | awk '{print$2}')
```

- e. Store the subnet ID as a local variable by running the following command:

```
$ export SUBNET_ID=$(aws cloudformation describe-stacks --stack-name "
<subnet_stack_name>" \ 1
| jq -r '.Stacks[0].Outputs[0].OutputValue')
```

- 1** For **<subnet_stack_name>**, specify the name of the subnet stack that you created.

- f. Store the cluster ID as local variable by running the following command:

```
$ export CLUSTER_ID="$(awk '/infrastructureName: / {print $2}'
<installation_directory>/manifests/cluster-infrastructure-02-config.yml)"
```

- g. Create the worker manifest file for the Local Zone that your VPC uses by running the following command:

```
$ cat <<EOF > <installation_directory>/openshift/99_openshift-cluster-api_worker-
machineset-nyc1.yaml
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: ${CLUSTER_ID}
    name: ${CLUSTER_ID}-edge-${LZ_ZONE_NAME}
    namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: ${CLUSTER_ID}
      machine.openshift.io/cluster-api-machineset: ${CLUSTER_ID}-edge-
${LZ_ZONE_NAME}
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: ${CLUSTER_ID}
        machine.openshift.io/cluster-api-machine-role: edge
        machine.openshift.io/cluster-api-machine-type: edge
        machine.openshift.io/cluster-api-machineset: ${CLUSTER_ID}-edge-
${LZ_ZONE_NAME}
    spec:
      metadata:
        labels:
          machine.openshift.com/zone-type: local-zone
          machine.openshift.com/zone-group: ${ZONE_GROUP_NAME}
          node-role.kubernetes.io/edge: ""
      taints:
        - key: node-role.kubernetes.io/edge
          effect: NoSchedule
      providerSpec:
```

```

value:
  ami:
    id: ${AMI_ID}
  apiVersion: machine.openshift.io/v1beta1
  blockDevices:
  - ebs:
    volumeSize: 120
    volumeType: gp2
  credentialsSecret:
    name: aws-cloud-credentials
  deviceIndex: 0
  iamInstanceProfile:
    id: ${CLUSTER_ID}-worker-profile
  instanceType: ${INSTANCE_TYPE}
  kind: AWSMachineProviderConfig
  placement:
    availabilityZone: ${LZ_ZONE_NAME}
    region: ${CLUSTER_REGION}
  securityGroups:
  - filters:
    - name: tag:Name
      values:
      - ${CLUSTER_ID}-worker-sg
  subnet:
    id: ${SUBNET_ID}
  publicIp: true
  tags:
  - name: kubernetes.io/cluster/${CLUSTER_ID}
    value: owned
  userDataSecret:
    name: worker-user-data

```

EOF

Additional resources

- [Changing the MTU for the cluster network](#)
- [Enabling IPsec encryption](#)

6.14.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



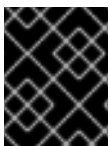
NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
```

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Next steps

- [Creating user workloads in AWS Local Zones](#)

6.14.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:


```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

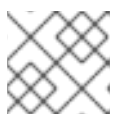
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

6.14.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.14.14. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

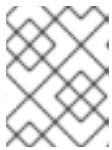


NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.14.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

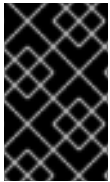
6.14.16. Next steps

- [Validating an installation.](#)

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

6.15. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

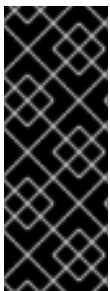
In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) using infrastructure that you provide and an internal mirror of the installation release content.



IMPORTANT

While you can install an OpenShift Container Platform cluster by using mirrored installation release content, your cluster still requires internet access to use the AWS APIs.

One way to create this infrastructure is to use the provided CloudFormation templates. You can modify the templates to customize your infrastructure or use the information that they contain to create AWS objects according to your company's policies.

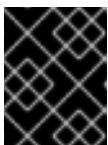


IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several CloudFormation templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

6.15.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a mirror registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You [configured an AWS account](#) to host the cluster.



IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#) in the AWS documentation.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.15.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

6.15.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.

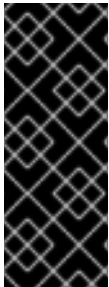
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

6.15.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.15.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

6.15.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 6.51. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

6.15.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.52. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.15.4.3. Tested instance types for AWS

The following Amazon Web Services (AWS) instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.53. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.15.4.4. Tested instance types for AWS on 64-bit ARM infrastructures

The following Amazon Web Services (AWS) ARM64 instance types have been tested with OpenShift Container Platform.



NOTE

Use the machine types included in the following charts for your AWS ARM instances. If you use an instance type that is not listed in the chart, ensure that the instance size you use matches the minimum resource requirements that are listed in "Minimum resource requirements for cluster installation".

Example 6.54. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

6.15.4.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

6.15.5. Required AWS infrastructure components

To install OpenShift Container Platform on user-provisioned infrastructure in Amazon Web Services (AWS), you must manually create both the machines and their supporting infrastructure.

For more information about the integration testing for different platforms, see the [OpenShift Container Platform 4.x Tested Integrations](#) page.

By using the provided CloudFormation templates, you can create stacks of AWS resources that represent the following components:

- An AWS Virtual Private Cloud (VPC)
- Networking and load balancing components
- Security groups and roles
- An OpenShift Container Platform bootstrap node
- OpenShift Container Platform control plane nodes
- An OpenShift Container Platform compute node

Alternatively, you can manually create the components or you can reuse existing infrastructure that meets the cluster requirements. Review the CloudFormation templates for more details about how the components interrelate.

6.15.5.1. Other infrastructure components

- A VPC

- DNS entries
- Load balancers (classic or network) and listeners
- A public and a private Route 53 zone
- Security groups
- IAM roles
- S3 buckets

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2, ELB, and S3 endpoints. Depending on the level to which you want to restrict internet traffic during the installation, the following configuration options are available:

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description

Component	AWS type	Description												
VPC	<ul style="list-style-type: none"> ● AWS::EC2::VPC ● AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.												
Public subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::SubnetNetworkAclAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.												
Internet gateway	<ul style="list-style-type: none"> ● AWS::EC2::InternetGateway ● AWS::EC2::VPCGatewayAttachment ● AWS::EC2::RouteTable ● AWS::EC2::Route ● AWS::EC2::SubnetRouteTableAssociation ● AWS::EC2::NatGateway ● AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.												
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	<p>You must allow the VPC to access the following ports:</p> <table border="1"> <thead> <tr> <th>Port</th> <th>Reason</th> </tr> </thead> <tbody> <tr> <td>80</td> <td>Inbound HTTP traffic</td> </tr> <tr> <td>443</td> <td>Inbound HTTPS traffic</td> </tr> <tr> <td>22</td> <td>Inbound SSH traffic</td> </tr> <tr> <td>1024 - 65535</td> <td>Inbound ephemeral traffic</td> </tr> <tr> <td>0 - 65535</td> <td>Outbound ephemeral traffic</td> </tr> </tbody> </table>	Port	Reason	80	Inbound HTTP traffic	443	Inbound HTTPS traffic	22	Inbound SSH traffic	1024 - 65535	Inbound ephemeral traffic	0 - 65535	Outbound ephemeral traffic
Port	Reason													
80	Inbound HTTP traffic													
443	Inbound HTTPS traffic													
22	Inbound SSH traffic													
1024 - 65535	Inbound ephemeral traffic													
0 - 65535	Outbound ephemeral traffic													

Component	AWS type	Description
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

Required DNS and load balancing components

Your DNS and load balancer configuration needs to use a public hosted zone and can use a private hosted zone similar to the one that the installation program uses if it provisions the cluster's infrastructure. You must create a DNS entry that resolves to your load balancer. An entry for **api.<cluster_name>.<domain>** must point to the external load balancer, and an entry for **api-int.<cluster_name>.<domain>** must point to the internal load balancer.

The cluster also requires load balancers and listeners for port 6443, which are required for the Kubernetes API and its extensions, and port 22623, which are required for the Ignition config files for new machines. The targets will be the control plane nodes. Port 6443 must be accessible to both clients external to the cluster and nodes within the cluster. Port 22623 must be accessible to nodes within the cluster.

Component	AWS type	Description
DNS	AWS::Route53::HostedZone	The hosted zone for your internal DNS.
Public load balancer	AWS::ElasticLoadBalancingV2::LoadBalancer	The load balancer for your public subnets.
External API server record	AWS::Route53::RecordSetGroup	Alias records for the external API server.
External listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 6443 for the external load balancer.
External target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the external load balancer.

Component	AWS type	Description
Private load balancer	AWS::ElasticLoadBalancingV2::LoadBalancer	The load balancer for your private subnets.
Internal API server record	AWS::Route53::RecordSetGroup	Alias records for the internal API server.
Internal listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 22623 for the internal load balancer.
Internal target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the internal load balancer.
Internal listener	AWS::ElasticLoadBalancingV2::Listener	A listener on port 6443 for the internal load balancer.
Internal target group	AWS::ElasticLoadBalancingV2::TargetGroup	The target group for the internal load balancer.

Security groups

The control plane and worker machines require access to the following ports:

Group	Type	IP Protocol	Port range
MasterSecurityGroup	AWS::EC2::SecurityGroup	icmp	0
		tcp	22
		tcp	6443
		tcp	22623
WorkerSecurityGroup	AWS::EC2::SecurityGroup	icmp	0

Group	Type	IP Protocol	Port range
		tcp	22
BootstrapSecurityGroup	AWS::EC2::SecurityGroup	tcp	22
		tcp	19531

Control plane Ingress

The control plane machines require the following Ingress groups. Each Ingress group is a **AWS::EC2::SecurityGroupIngress** resource.

Ingress group	Description	IP protocol	Port range
MasterIngress Etcd	etcd	tcp	2379- 2380
MasterIngress Vxlan	Vxlan packets	udp	4789
MasterIngress WorkerVxlan	Vxlan packets	udp	4789
MasterIngress Internal	Internal cluster communication and Kubernetes proxy metrics	tcp	9000 - 9999
MasterIngress WorkerInternal	Internal cluster communication	tcp	9000 - 9999
MasterIngress Kube	Kubernetes kubelet, scheduler and controller manager	tcp	10250 - 10259
MasterIngress WorkerKube	Kubernetes kubelet, scheduler and controller manager	tcp	10250 - 10259
MasterIngress IngressServices	Kubernetes Ingress services	tcp	30000 - 32767
MasterIngress WorkerIngressServices	Kubernetes Ingress services	tcp	30000 - 32767
MasterIngress Geneve	Geneve packets	udp	6081

Ingress group	Description	IP protocol	Port range
MasterIngress WorkerGeneve	Geneve packets	udp	6081
MasterIngress IpsecIke	IPsec IKE packets	udp	500
MasterIngress WorkerIpsecIke	IPsec IKE packets	udp	500
MasterIngress IpsecNat	IPsec NAT-T packets	udp	4500
MasterIngress WorkerIpsecNat	IPsec NAT-T packets	udp	4500
MasterIngress IpsecEsp	IPsec ESP packets	50	All
MasterIngress WorkerIpsecEsp	IPsec ESP packets	50	All
MasterIngress InternalUDP	Internal cluster communication	udp	9000 - 9999
MasterIngress WorkerInternalUDP	Internal cluster communication	udp	9000 - 9999
MasterIngress IngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767
MasterIngress WorkerIngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767

Worker Ingress

The worker machines require the following Ingress groups. Each Ingress group is a **AWS::EC2::SecurityGroupIngress** resource.

Ingress group	Description	IP protocol	Port range
WorkerIngress Vxlan	Vxlan packets	udp	4789
WorkerIngress WorkerVxlan	Vxlan packets	udp	4789
WorkerIngress Internal	Internal cluster communication	tcp	9000 - 9999
WorkerIngress WorkerInternal	Internal cluster communication	tcp	9000 - 9999
WorkerIngress Kube	Kubernetes kubelet, scheduler, and controller manager	tcp	10250
WorkerIngress WorkerKube	Kubernetes kubelet, scheduler, and controller manager	tcp	10250
WorkerIngress IngressServices	Kubernetes Ingress services	tcp	30000 - 32767
WorkerIngress WorkerIngressServices	Kubernetes Ingress services	tcp	30000 - 32767
WorkerIngress Geneve	Geneve packets	udp	6081
WorkerIngress MasterGeneve	Geneve packets	udp	6081
WorkerIngress IpsecIke	IPsec IKE packets	udp	500
WorkerIngress MasterIpsecIke	IPsec IKE packets	udp	500
WorkerIngress IpsecNat	IPsec NAT-T packets	udp	4500
WorkerIngress MasterIpsecNat	IPsec NAT-T packets	udp	4500

Ingress group	Description	IP protocol	Port range
WorkerIngress IpsecEsp	IPsec ESP packets	50	All
WorkerIngress MasterIpsecEsp	IPsec ESP packets	50	All
WorkerIngress InternalUDP	Internal cluster communication	udp	9000 - 9999
WorkerIngress MasterInternal UDP	Internal cluster communication	udp	9000 - 9999
WorkerIngress IngressServicesUDP	Kubernetes Ingress services	udp	30000 - 32767
WorkerIngress MasterIngress ServicesUDP	Kubernetes Ingress services	udp	30000 - 32767

Roles and instance profiles

You must grant the machines permissions in AWS. The provided CloudFormation templates grant the machines **Allow** permissions for the following **AWS::IAM::Role** objects and provide a **AWS::IAM::InstanceProfile** for each set of roles. If you do not use the templates, you can grant the machines the following broad permissions or the following individual permissions.

Role	Effect	Action	Resource
Master	Allow	ec2:*	*
	Allow	elasticloadbalancing :*	*
	Allow	iam:PassRole	*
	Allow	s3:GetObject	*
Worker	Allow	ec2:Describe*	*
Bootstrap	Allow	ec2:Describe*	*
	Allow	ec2:AttachVolume	*

Role	Effect	Action	Resource
	Allow	ec2:DetachVolume	*

6.15.5.2. Cluster machines

You need **AWS::EC2::Instance** objects for the following machines:

- A bootstrap machine. This machine is required during installation, but you can remove it after your cluster deploys.
- Three control plane machines. The control plane machines are not governed by a control plane machine set.
- Compute machines. You must create at least two compute machines, which are also known as worker machines, during installation. These machines are not governed by a compute machine set.

6.15.5.3. Required AWS permissions for the IAM user



NOTE

Your IAM user must have the permission **tag:GetResources** in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

When you attach the **AdministratorAccess** policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

Example 6.55. Required EC2 permissions for installation

- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteSnapshot**
- **ec2>DeleteTags**

- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInstanceTypes**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**

- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

Example 6.56. Required permissions for creating network resources during installation

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



NOTE

If you use an existing VPC, your account does not require these permissions for creating network resources.

Example 6.57. Required Elastic Load Balancing permissions (ELB) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**

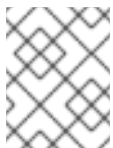
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing>CreateLoadBalancer**
- **elasticloadbalancing>CreateLoadBalancerListeners**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

Example 6.58. Required Elastic Load Balancing permissions (ELBv2) for installation

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing>CreateListener**
- **elasticloadbalancing>CreateLoadBalancer**
- **elasticloadbalancing>CreateTargetGroup**
- **elasticloadbalancing>DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**

Example 6.59. Required IAM permissions for installation

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam>DeleteRole**
- **iam>DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**

**NOTE**

If you have not created a load balancer in your AWS account, the IAM user also requires the **iam:CreateServiceLinkedRole** permission.

Example 6.60. Required Route 53 permissions for installation

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53>DeleteHostedZone**
- **route53:GetChange**

- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

Example 6.61. Required S3 permissions for installation

- **s3:CreateBucket**
- **s3>DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketPolicy**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

Example 6.62. S3 permissions that cluster Operators require

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

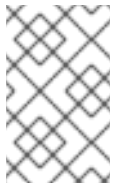
Example 6.63. Required permissions to delete base cluster resources

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeletePlacementGroup**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam>ListAttachedRolePolicies**
- **iam>ListInstanceProfiles**
- **iam>ListRolePolicies**
- **iam>ListUserPolicies**
- **s3:DeleteObject**
- **s3>ListBucketVersions**
- **tag:GetResources**

Example 6.64. Required permissions to delete network resources

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**

- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **ec2:ReplaceRouteTableAssociation**



NOTE

If you use an existing VPC, your account does not require these permissions to delete network resources. Instead, your account only requires the **tag:UntagResources** permission to delete network resources.

Example 6.65. Required permissions to delete a cluster with shared instance roles

- **iam:UntagRole**

Example 6.66. Additional IAM and S3 permissions that are required to create manifests

- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:ListBucket**

- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**



NOTE

If you are managing your cloud provider credentials with mint mode, the IAM user also requires the **iam:CreateAccessKey** and **iam:CreateUser** permissions.

Example 6.67. Optional permissions for instance and quota checks for installation

- **ec2:DescribeInstanceTypeOfferings**
- **servicequotas:ListAWSDefaultServiceQuotas**

6.15.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

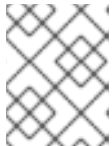
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

6.15.7. Creating the installation files for AWS

To install OpenShift Container Platform on Amazon Web Services (AWS) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

6.15.7.1. Optional: Creating a separate **/var** partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

- Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

- Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> ❶
      partitions:
        - label: var
          start_mib: <partition_start_offset> ❷
          size_mib: <partition_size> ❸
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] ❹
          with_mount_unit: true
```

- The storage device name of the disk that you want to partition.
- When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future

reinstalls of RHCOS might overwrite the beginning of the data partition.

- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

6.15.7.2. Creating the installation configuration file

Generate and customize the installation configuration file that the installation program needs to deploy your cluster.

Prerequisites

- You obtained the OpenShift Container Platform installation program for user-provisioned infrastructure and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- You checked that you are deploying your cluster to a region with an accompanying Red Hat Enterprise Linux CoreOS (RHCOS) AMI published by Red Hat. If you are deploying to a region that requires a custom AMI, such as an AWS GovCloud region, you must create the **install-config.yaml** file manually.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.



IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **aws** as the platform to target.
- iii. If you do not have an AWS profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.



NOTE

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

- iv. Select the AWS region to deploy the cluster to.
 - v. Select the base domain for the Route 53 service that you configured for your cluster.
 - vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}'
```

For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value. The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----END CERTIFICATE-----
```

- c. Add the image content resources:

```
imageContentSources:
- mirrors:
  - <local_registry>/<local_repository_name>/release
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

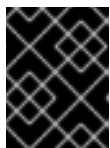
Use the **imageContentSources** section from the output of the command to mirror the repository or the values that you used when you mirrored the content from the media that you brought into your restricted network.

- d. Optional: Set the publishing strategy to **Internal**:

```
publish: Internal
```

By setting this option, you create an internal Ingress Controller and a private load balancer.

3. Optional: Back up the **install-config.yaml** file.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- See [Configuration and credential file settings](#) in the AWS documentation for more information about AWS profile and credential configuration.

6.15.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

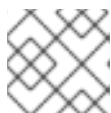
Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. If you have added the Amazon **EC2**, **Elastic Load Balancing**, and **S3** VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

6.15.7.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the control plane machine set:

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
```

```
privateZone: ❶
  id: mycluster-100419-private-zone
publicZone: ❷
  id: example.openshift.com
status: {}
```

- ❶ ❷ Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. Optional: If you manually created a cloud identity and access management (IAM) role, locate any **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation in the release image by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=<platform_name>
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```



IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

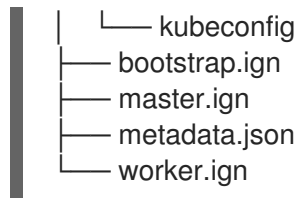
- a. Delete all **CredentialsRequest** objects that have the **TechPreviewNoUpgrade** annotation.
7. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> ❶
```

- ❶ For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   └── kubeadmin-password
```



Additional resources

- [Manually creating IAM](#)

6.15.8. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Amazon Web Services (AWS). The infrastructure name is also used to locate the appropriate AWS resources during an OpenShift Container Platform installation. The provided CloudFormation templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

6.15.9. Creating a VPC in AWS

You must create a Virtual Private Cloud (VPC) in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements, including VPN and route tables.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the VPC.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.

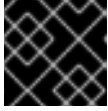
Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

- 1** The CIDR block for the VPC.
- 2** Specify a CIDR block in the format **x.x.x.x/16-24**.
- 3** The number of availability zones to deploy the VPC in.
- 4** Specify an integer between **1** and **3**.
- 5** The size of each subnet in each availability zone.
- 6** Specify an integer between **5** and **13**, where **5** is **/27** and **13** is **/19**.

2. Copy the template from the **CloudFormation template for the VPC** section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.
3. Launch the CloudFormation template to create a stack of AWS resources that represent the VPC:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
```

- 1** **<name>** is the name for the CloudFormation stack, such as **cluster-vpc**. You need the name of this stack if you remove the cluster.
- 2** **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3** **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

VpcId	The ID of your VPC.
PublicSubnetIds	The IDs of the new public subnets.
PrivateSubnetIds	The IDs of the new private subnets.

6.15.9.1. CloudFormation template for the VPC

You can use the following CloudFormation template to deploy the VPC that you need for your OpenShift Container Platform cluster.

Example 6.68. CloudFormation template for the VPC

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
```

```

AllowedPattern: ^(((0-9){1-9}|0-9|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}((0-9){1-9}|0-9|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.(1[6-9]|2[0-4]))$
ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
Default: 10.0.0.0/16
Description: CIDR block for VPC.
Type: String
AvailabilityZoneCount:
ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
MinValue: 1
MaxValue: 3
Default: 1
Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
Type: Number
SubnetBits:
ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
MinValue: 5
MaxValue: 13
Default: 12
Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
Type: Number

Metadata:
AWS::CloudFormation::Interface:
ParameterGroups:
- Label:
  default: "Network Configuration"
Parameters:
- VpcCidr
- SubnetBits
- Label:
  default: "Availability Zones"
Parameters:
- AvailabilityZoneCount
ParameterLabels:
AvailabilityZoneCount:
  default: "Availability Zone Count"
VpcCidr:
  default: "VPC CIDR"
SubnetBits:
  default: "Bits Per Subnet"

Conditions:
DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
VPC:
Type: "AWS::EC2::VPC"
Properties:
  EnableDnsSupport: "true"
  EnableDnsHostnames: "true"
  CidrBlock: !Ref VpcCidr
PublicSubnet:
Type: "AWS::EC2::Subnet"
Properties:

```



```

VpcId: !Ref VPC
CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
AvailabilityZone: !Select
- 0
- Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet2:
Type: "AWS::EC2::Subnet"
Condition: DoAz2
Properties:
VpcId: !Ref VPC
CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
AvailabilityZone: !Select
- 1
- Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet3:
Type: "AWS::EC2::Subnet"
Condition: DoAz3
Properties:
VpcId: !Ref VPC
CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
AvailabilityZone: !Select
- 2
- Fn::GetAZs: !Ref "AWS::Region"
InternetGateway:
Type: "AWS::EC2::InternetGateway"
GatewayToInternet:
Type: "AWS::EC2::VPCGatewayAttachment"
Properties:
VpcId: !Ref VPC
InternetGatewayId: !Ref InternetGateway
PublicRouteTable:
Type: "AWS::EC2::RouteTable"
Properties:
VpcId: !Ref VPC
PublicRoute:
Type: "AWS::EC2::Route"
DependsOn: GatewayToInternet
Properties:
RouteTableId: !Ref PublicRouteTable
DestinationCidrBlock: 0.0.0.0/0
GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
Type: "AWS::EC2::SubnetRouteTableAssociation"
Properties:
SubnetId: !Ref PublicSubnet
RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
Type: "AWS::EC2::SubnetRouteTableAssociation"
Condition: DoAz2
Properties:
SubnetId: !Ref PublicSubnet2
RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
Condition: DoAz3
Type: "AWS::EC2::SubnetRouteTableAssociation"
Properties:

```

```
SubnetId: !Ref PublicSubnet3
RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP
        - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
```

```

Type: "AWS::EC2::SubnetRouteTableAssociation"
Condition: DoAz2
Properties:
  SubnetId: !Ref PrivateSubnet2
  RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:

```

```

AllocationId:
  "Fn::GetAtt":
    - EIP3
    - AllocationId
SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Principal: '*'
          Action:
            - '*'
          Resource:
            - '*'
    RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
    ServiceName: !Join
      - "
      - - com.amazonaws.
      - - !Ref 'AWS::Region'
      - - .s3
    Vpclid: !Ref VPC

Outputs:
Vpclid:
  Description: ID of the new VPC.
  Value: !Ref VPC
PublicSubnetIds:
  Description: Subnet IDs of the public subnets.
  Value:
    !Join [
      " ",
      [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
    ]
PrivateSubnetIds:

```

Description: Subnet IDs of the private subnets.

Value:

```
!Join [
  ",",
  [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
]
```

6.15.10. Creating networking and load balancing components in AWS

You must configure networking and classic or network load balancing in Amazon Web Services (AWS) that your OpenShift Container Platform cluster can use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the networking and load balancing components that your OpenShift Container Platform cluster requires. The template also creates a hosted zone and subnet tags.

You can run the template multiple times within a single Virtual Private Cloud (VPC).



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

Procedure

1. Obtain the hosted zone ID for the Route 53 base domain that you specified in the **install-config.yaml** file for your cluster. You can obtain details about your hosted zone by running the following command:

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

- 1** For the **<route53_domain>**, specify the Route 53 base domain that you used when you generated the **install-config.yaml** file for the cluster.

Example output

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

In the example output, the hosted zone ID is **Z21IXYZABCZ2A4**.

2. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

- 1 A short, representative cluster name to use for hostnames, etc.
- 2 Specify the cluster name that you used when you generated the **install-config.yaml** file for the cluster.
- 3 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 4 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 5 The Route 53 public zone ID to register the targets with.
- 6 Specify the Route 53 public zone ID, which has a format similar to **Z21IXYZABCZ2A4**. You can obtain this value from the AWS console.

- 7 The Route 53 zone to register the targets with.
 - 8 Specify the Route 53 base domain that you used when you generated the **install-config.yaml** file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
 - 9 The public subnets that you created for your VPC.
 - 10 Specify the **PublicSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 11 The private subnets that you created for your VPC.
 - 12 Specify the **PrivateSubnetIds** value from the output of the CloudFormation template for the VPC.
 - 13 The VPC that you created for the cluster.
 - 14 Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
3. Copy the template from the **CloudFormation template for the network and load balancers** section of this topic and save it as a YAML file on your computer. This template describes the networking and load balancing objects that your cluster requires.



IMPORTANT

If you are deploying your cluster to an AWS government or secret region, you must update the **InternalApiServerRecord** in the CloudFormation template to use **CNAME** records. Records of type **ALIAS** are not supported for AWS government regions.

4. Launch the CloudFormation template to create a stack of AWS resources that provide the networking and load balancing components:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-dns**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

4

You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

PrivateHostedZoneId	Hosted zone ID for the private DNS.
ExternalApiLoadBalancerName	Full name of the external API load balancer.
InternalApiLoadBalancerName	Full name of the internal API load balancer.
ApiServerDnsName	Full hostname of the API server.
RegisterNlbTargetsLambda	Lambda ARN useful to help register/deregister IP targets for these load balancers.
ExternalApiTargetGroupArn	ARN of external API target group.
InternalApiTargetGroupArn	ARN of internal API target group.
InternalServiceTargetGroupArn	ARN of internal service target group.

6.15.10.1. CloudFormation template for the network and load balancers

You can use the following CloudFormation template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster.

Example 6.69. CloudFormation template for the network and load balancers

```

AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
  ClusterName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
    maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
    names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
    maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
    used by the cluster.
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as
    Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
    Description: The Route53 zone to register the targets with, such as example.com. Omit the
    trailing period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Cluster Information"
        Parameters:
          - ClusterName
          - InfrastructureName
      - Label:
          default: "Network Configuration"

```

```
Parameters:
- VpcId
- PublicSubnets
- PrivateSubnets
- Label:
  default: "DNS"
Parameters:
- HostedZoneName
- HostedZoneId
ParameterLabels:
ClusterName:
  default: "Cluster Name"
InfrastructureName:
  default: "Infrastructure Name"
VpcId:
  default: "VPC ID"
PublicSubnets:
  default: "Public Subnets"
PrivateSubnets:
  default: "Private Subnets"
HostedZoneName:
  default: "Public Hosted Zone Name"
HostedZoneId:
  default: "Public Hosted Zone ID"
```

Resources:

```
ExtApiElb:
Type: AWS::ElasticLoadBalancingV2::LoadBalancer
Properties:
Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
IpAddressType: ipv4
Subnets: !Ref PublicSubnets
Type: network
```

```
IntApiElb:
Type: AWS::ElasticLoadBalancingV2::LoadBalancer
Properties:
Name: !Join ["-", [!Ref InfrastructureName, "int"]]
Scheme: internal
IpAddressType: ipv4
Subnets: !Ref PrivateSubnets
Type: network
```

```
IntDns:
Type: "AWS::Route53::HostedZone"
Properties:
HostedZoneConfig:
  Comment: "Managed by CloudFormation"
Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
HostedZoneTags:
- Key: Name
  Value: !Join ["-", [!Ref InfrastructureName, "int"]]
- Key: !Join ["/", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
  Value: "owned"
VPCs:
- VPCId: !Ref VpcId
```

VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:

Type: AWS::Route53::RecordSetGroup

Properties:

Comment: Alias record for the API server

HostedZoneId: !Ref HostedZoneId

RecordSets:

- Name:

```
!Join [
  ":",
  ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]],
]
```

Type: A

AliasTarget:

HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID

DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:

Type: AWS::Route53::RecordSetGroup

Properties:

Comment: Alias record for the API server

HostedZoneId: !Ref IntDns

RecordSets:

- Name:

```
!Join [
  ":",
  ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]],
]
```

Type: A

AliasTarget:

HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID

DNSName: !GetAtt IntApiElb.DNSName

- Name:

```
!Join [
  ":",
  ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]],
]
```

Type: A

AliasTarget:

HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID

DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:

Type: AWS::ElasticLoadBalancingV2::Listener

Properties:

DefaultActions:

- Type: forward

TargetGroupArn:

Ref: ExternalApiTargetGroup

LoadBalancerArn:

Ref: ExtApiElb

Port: 6443

Protocol: TCP

ExternalApiTargetGroup:

Type: AWS::ElasticLoadBalancingV2::TargetGroup

Properties:

HealthCheckIntervalSeconds: 10

HealthCheckPath: "/readyz"

HealthCheckPort: 6443

HealthCheckProtocol: HTTPS

HealthyThresholdCount: 2

UnhealthyThresholdCount: 2

Port: 6443

Protocol: TCP

TargetType: ip

VpcId:

Ref: VpcId

TargetGroupAttributes:

- Key: deregistration_delay.timeout_seconds

Value: 60

InternalApiListener:

Type: AWS::ElasticLoadBalancingV2::Listener

Properties:

DefaultActions:

- Type: forward

TargetGroupArn:

Ref: InternalApiTargetGroup

LoadBalancerArn:

Ref: IntApiElb

Port: 6443

Protocol: TCP

InternalApiTargetGroup:

Type: AWS::ElasticLoadBalancingV2::TargetGroup

Properties:

HealthCheckIntervalSeconds: 10

HealthCheckPath: "/readyz"

HealthCheckPort: 6443

HealthCheckProtocol: HTTPS

HealthyThresholdCount: 2

UnhealthyThresholdCount: 2

Port: 6443

Protocol: TCP

TargetType: ip

VpcId:

Ref: VpcId

TargetGroupAttributes:

- Key: deregistration_delay.timeout_seconds

Value: 60

InternalServiceInternalListener:

Type: AWS::ElasticLoadBalancingV2::Listener

Properties:

DefaultActions:

- Type: forward

TargetGroupArn:

Ref: InternalServiceTargetGroup

LoadBalancerArn:

Ref: IntApiElb

Port: 22623
 Protocol: TCP

InternalServiceTargetGroup:

Type: AWS::ElasticLoadBalancingV2::TargetGroup

Properties:

HealthCheckIntervalSeconds: 10

HealthCheckPath: "/healthz"

HealthCheckPort: 22623

HealthCheckProtocol: HTTPS

HealthyThresholdCount: 2

UnhealthyThresholdCount: 2

Port: 22623

Protocol: TCP

TargetType: ip

VpcId:

Ref: VpcId

TargetGroupAttributes:

- Key: deregistration_delay.timeout_seconds
 Value: 60

RegisterTargetLambdalamRole:

Type: AWS::IAM::Role

Properties:

RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "lambda.amazonaws.com"

Action:

- "sts:AssumeRole"

Path: "/"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action:

```
[
  "elasticloadbalancing:RegisterTargets",
  "elasticloadbalancing:DeregisterTargets",
]
```

Resource: !Ref InternalApiTargetGroup

- Effect: "Allow"

Action:

```
[
  "elasticloadbalancing:RegisterTargets",
  "elasticloadbalancing:DeregisterTargets",
]
```

Resource: !Ref InternalServiceTargetGroup

- Effect: "Allow"

Action:

```

    [
      "elasticloadbalancing:RegisterTargets",
      "elasticloadbalancing:DeregisterTargets",
    ]
    Resource: !Ref ExternalApiTargetGroup

```

RegisterNlbPTargets:

```

Type: "AWS::Lambda::Function"
Properties:
  Handler: "index.handler"
  Role:
    Fn::GetAtt:
      - "RegisterTargetLambdalamRole"
      - "Arn"
  Code:
    ZipFile: |
      import json
      import boto3
      import cfnresponse
      def handler(event, context):
        elb = boto3.client('elbv2')
        if event['RequestType'] == 'Delete':
          elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
[TargetArn],Targets=[{'Id': event['ResourceProperties']['TargetIp']})
          elif event['RequestType'] == 'Create':
            elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetIp']})
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
  Runtime: "python3.8"
  Timeout: 120

```

RegisterSubnetTagsLambdalamRole:

```

Type: AWS::IAM::Role
Properties:
  RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
  AssumeRolePolicyDocument:
    Version: "2012-10-17"
    Statement:
      - Effect: "Allow"
        Principal:
          Service:
            - "lambda.amazonaws.com"
        Action:
          - "sts:AssumeRole"
  Path: "/"
  Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",

```

```

    "ec2:CreateTags"
  ]
  Resource: "arn:aws:ec2:*:*:subnet/*"
- Effect: "Allow"
  Action:
  [
    "ec2:DescribeSubnets",
    "ec2:DescribeTags"
  ]
  Resource: "*"

```

RegisterSubnetTags:

Type: "AWS::Lambda::Function"

Properties:

Handler: "index.handler"

Role:

Fn::GetAtt:

- "RegisterSubnetTagsLambdalamRole"

- "Arn"

Code:

ZipFile: |

```

import json
import boto3
import cfnresponse
def handler(event, context):
    ec2_client = boto3.client('ec2')
    if event['RequestType'] == 'Delete':
        for subnet_id in event['ResourceProperties']['Subnets']:
            ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
    elif event['RequestType'] == 'Create':
        for subnet_id in event['ResourceProperties']['Subnets']:
            ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
    responseData = {}
    cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
Runtime: "python3.8"
Timeout: 120

```

RegisterPublicSubnetTags:

Type: Custom::SubnetRegister

Properties:

ServiceToken: !GetAtt RegisterSubnetTags.Arn

InfrastructureName: !Ref InfrastructureName

Subnets: !Ref PublicSubnets

RegisterPrivateSubnetTags:

Type: Custom::SubnetRegister

Properties:

ServiceToken: !GetAtt RegisterSubnetTags.Arn

InfrastructureName: !Ref InfrastructureName

Subnets: !Ref PrivateSubnets

Outputs:

PrivateHostedZoneId:

Description: Hosted zone ID for the private DNS, which is required for private records.
 Value: `!Ref IntDns`

ExternalApiLoadBalancerName:
 Description: Full name of the external API load balancer.
 Value: `!GetAtt ExtApiElb.LoadBalancerFullName`

InternalApiLoadBalancerName:
 Description: Full name of the internal API load balancer.
 Value: `!GetAtt IntApiElb.LoadBalancerFullName`

ApiServerDnsName:
 Description: Full hostname of the API server, which is required for the Ignition config files.
 Value: `!Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]`

RegisterNlbTargetsLambda:
 Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.
 Value: `!GetAtt RegisterNlbTargets.Arn`

ExternalApiTargetGroupArn:
 Description: ARN of the external API target group.
 Value: `!Ref ExternalApiTargetGroup`

InternalApiTargetGroupArn:
 Description: ARN of the internal API target group.
 Value: `!Ref InternalApiTargetGroup`

InternalServiceTargetGroupArn:
 Description: ARN of the internal service target group.
 Value: `!Ref InternalServiceTargetGroup`

IMPORTANT

If you are deploying your cluster to an AWS government or secret region, you must update the **InternalApiServerRecord** to use **CNAME** records. Records of type **ALIAS** are not supported for AWS government regions. For example:

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

Additional resources

- See [Listing public hosted zones](#) in the AWS documentation for more information about listing public hosted zones.

6.15.11. Creating security group and roles in AWS

You must create security groups and roles in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the security groups and roles that your OpenShift Container Platform cluster requires.



NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

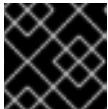
Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

- 1** The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2** Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3** The CIDR block for the VPC.
- 4** Specify the CIDR block parameter that you used for the VPC that you defined in the form **x.x.x.x/16-24**.
- 5** The private subnets that you created for your VPC.
- 6** Specify the **PrivateSubnetIds** value from the output of the CloudFormation template for the VPC.

- 7 The VPC that you created for the cluster.
 - 8 Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
2. Copy the template from the **CloudFormation template for security objects** section of this topic and save it as a YAML file on your computer. This template describes the security groups and roles that your cluster requires.
 3. Launch the CloudFormation template to create a stack of AWS resources that represent the security groups and roles:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-sec**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.
- 4 You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** and **AWS::IAM::InstanceProfile** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

MasterSecurityGroupID	Master Security Group ID
------------------------------	--------------------------

WorkerSecurityGroupID	Worker Security Group ID
MasterInstanceProfile	Master IAM Instance Profile
WorkerInstanceProfile	Worker IAM Instance Profile

6.15.11.1. CloudFormation template for security objects

You can use the following CloudFormation template to deploy the security objects that you need for your OpenShift Container Platform cluster.

Example 6.70. CloudFormation template for security objects

```

AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^((([0-9]{1-9}[0-9]{1-9}|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]{1-9}[0-9]{1-9}|1[0-9]{2}|2[0-
4][0-9]|25[0-5])|(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Cluster Information"

```

Parameters:
- InfrastructureName
- Label:
 default: "Network Configuration"
Parameters:
- VpcId
- VpcCidr
- PrivateSubnets
ParameterLabels:
InfrastructureName:
 default: "Infrastructure Name"
VpcId:
 default: "VPC ID"
VpcCidr:
 default: "VPC CIDR"
PrivateSubnets:
 default: "Private Subnets"

Resources:
MasterSecurityGroup:
 Type: AWS::EC2::SecurityGroup
 Properties:
 GroupDescription: Cluster Master Security Group
 SecurityGroupIngress:
 - IpProtocol: icmp
 FromPort: 0
 ToPort: 0
 CidrIp: !Ref VpcCidr
 - IpProtocol: tcp
 FromPort: 22
 ToPort: 22
 CidrIp: !Ref VpcCidr
 - IpProtocol: tcp
 ToPort: 6443
 FromPort: 6443
 CidrIp: !Ref VpcCidr
 - IpProtocol: tcp
 FromPort: 22623
 ToPort: 22623
 CidrIp: !Ref VpcCidr
 VpcId: !Ref VpcId

WorkerSecurityGroup:
 Type: AWS::EC2::SecurityGroup
 Properties:
 GroupDescription: Cluster Worker Security Group
 SecurityGroupIngress:
 - IpProtocol: icmp
 FromPort: 0
 ToPort: 0
 CidrIp: !Ref VpcCidr
 - IpProtocol: tcp
 FromPort: 22
 ToPort: 22
 CidrIp: !Ref VpcCidr
 VpcId: !Ref VpcId

MasterIngressEtcD:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: etcd

FromPort: 2379

ToPort: 2380

IpProtocol: tcp

MasterIngressVxlan:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Vxlan packets

FromPort: 4789

ToPort: 4789

IpProtocol: udp

MasterIngressWorkerVxlan:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Vxlan packets

FromPort: 4789

ToPort: 4789

IpProtocol: udp

MasterIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Geneve packets

FromPort: 6081

ToPort: 6081

IpProtocol: udp

MasterIngressWorkerGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Geneve packets

FromPort: 6081

ToPort: 6081

IpProtocol: udp

MasterIngressIpsecIke:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec IKE packets

FromPort: 500

ToPort: 500

IpProtocol: udp

MasterIngressIpsecNat:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec NAT-T packets

FromPort: 4500

ToPort: 4500

IpProtocol: udp

MasterIngressIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec ESP packets

IpProtocol: 50

MasterIngressWorkerIpsecIke:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec IKE packets

FromPort: 500

ToPort: 500

IpProtocol: udp

MasterIngressWorkerIpsecNat:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec NAT-T packets

FromPort: 4500

ToPort: 4500

IpProtocol: udp

MasterIngressWorkerIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: IPsec ESP packets

IpProtocol: 50

MasterIngressInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

MasterIngressWorkerInternal:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

MasterIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

MasterIngressWorkerInternalUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

MasterIngressKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes kubelet, scheduler and controller manager
FromPort: 10250
ToPort: 10259
IpProtocol: tcp

MasterIngressWorkerKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes kubelet, scheduler and controller manager
FromPort: 10250
ToPort: 10259
IpProtocol: tcp

MasterIngressIngressServices:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: tcp

MasterIngressWorkerIngressServices:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: tcp

MasterIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

WorkerIngressVxlan:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Vxlan packets

FromPort: 4789

ToPort: 4789

IpProtocol: udp

WorkerIngressMasterVxlan:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Vxlan packets

FromPort: 4789

ToPort: 4789
IpProtocol: udp

WorkerIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

WorkerIngressMasterGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

WorkerIngressIpsecIke:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: IPsec IKE packets
FromPort: 500
ToPort: 500
IpProtocol: udp

WorkerIngressIpsecNat:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: IPsec NAT-T packets
FromPort: 4500
ToPort: 4500
IpProtocol: udp

WorkerIngressIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: IPsec ESP packets
IpProtocol: 50

WorkerIngressMasterIpsecIke:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec IKE packets

FromPort: 500

ToPort: 500

IpProtocol: udp

WorkerIngressMasterIpsecNat:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec NAT-T packets

FromPort: 4500

ToPort: 4500

IpProtocol: udp

WorkerIngressMasterIpsecEsp:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: IPsec ESP packets

IpProtocol: 50

WorkerIngressInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: tcp

WorkerIngressMasterInternal:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: tcp

WorkerIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

WorkerIngressMasterInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

WorkerIngressKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes secure kubelet port
FromPort: 10250
ToPort: 10250
IpProtocol: tcp

WorkerIngressWorkerKube:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal Kubernetes communication
FromPort: 10250
ToPort: 10250
IpProtocol: tcp

WorkerIngressIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

WorkerIngressMasterIngressServices:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

WorkerIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IProtocol: udp

MasterIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action:

- "ec2:AttachVolume"

- "ec2:AuthorizeSecurityGroupIngress"

- "ec2:CreateSecurityGroup"

- "ec2:CreateTags"

- "ec2:CreateVolume"

- "ec2>DeleteSecurityGroup"

- "ec2>DeleteVolume"

- "ec2:Describe*"

- "ec2:DetachVolume"

- "ec2:ModifyInstanceAttribute"

- "ec2:ModifyVolume"

- "ec2:RevokeSecurityGroupIngress"

- "elasticloadbalancing:AddTags"

- "elasticloadbalancing:AttachLoadBalancerToSubnets"

- "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"

- "elasticloadbalancing:CreateListener"

- "elasticloadbalancing:CreateLoadBalancer"

- "elasticloadbalancing:CreateLoadBalancerPolicy"

- "elasticloadbalancing:CreateLoadBalancerListeners"

- "elasticloadbalancing:CreateTargetGroup"

- "elasticloadbalancing:ConfigureHealthCheck"

- "elasticloadbalancing>DeleteListener"

- "elasticloadbalancing>DeleteLoadBalancer"

- "elasticloadbalancing>DeleteLoadBalancerListeners"

- "elasticloadbalancing>DeleteTargetGroup"

- "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"

- "elasticloadbalancing:DeregisterTargets"
- "elasticloadbalancing:Describe*"
- "elasticloadbalancing:DetachLoadBalancerFromSubnets"
- "elasticloadbalancing:ModifyListener"
- "elasticloadbalancing:ModifyLoadBalancerAttributes"
- "elasticloadbalancing:ModifyTargetGroup"
- "elasticloadbalancing:ModifyTargetGroupAttributes"
- "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
- "elasticloadbalancing:RegisterTargets"
- "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
- "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
- "kms:DescribeKey"

Resource: ""

MasterInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Roles:

- Ref: "MasterIamRole"

WorkerIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action:

- "ec2:DescribeInstances"

- "ec2:DescribeRegions"

Resource: ""

WorkerInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Roles:

- Ref: "WorkerIamRole"

Outputs:

MasterSecurityGroupId:

Description: Master Security Group ID

Value: !GetAtt MasterSecurityGroup.GroupId

WorkerSecurityGroupId:

Description: Worker Security Group ID

```
Value: !GetAtt WorkerSecurityGroup.GroupId
```

```
MasterInstanceProfile:
```

```
Description: Master IAM Instance Profile
```

```
Value: !Ref MasterInstanceProfile
```

```
WorkerInstanceProfile:
```

```
Description: Worker IAM Instance Profile
```

```
Value: !Ref WorkerInstanceProfile
```

6.15.12. Accessing RHCOS AMIs with stream metadata

In OpenShift Container Platform, *stream metadata* provides standardized metadata about RHCOS in the JSON format and injects the metadata into the cluster. Stream metadata is a stable format that supports multiple architectures and is intended to be self-documenting for maintaining automation.

You can use the **coreos print-stream-json** sub-command of **openshift-install** to access information about the boot images in the stream metadata format. This command provides a method for printing stream metadata in a scriptable, machine-readable format.

For user-provisioned installations, the **openshift-install** binary contains references to the version of RHCOS boot images that are tested for use with OpenShift Container Platform, such as the AWS AMI.

Procedure

To parse the stream metadata, use one of the following methods:

- From a Go program, use the official **stream-metadata-go** library at <https://github.com/coreos/stream-metadata-go>. You can also view example code in the library.
- From another programming language, such as Python or Ruby, use the JSON library of your preferred programming language.
- From a command-line utility that handles JSON data, such as **jq**:
 - Print the current **x86_64** or **aarch64** AMI for an AWS region, such as **us-west-1**:

For x86_64

```
$ openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.images.aws.regions["us-west-1"].image'
```

Example output

```
ami-0d3e625f84626bbda
```

For aarch64

```
$ openshift-install coreos print-stream-json | jq -r
'.architectures.aarch64.images.aws.regions["us-west-1"].image'
```

Example output

```
ami-0af1d3b7fa5be2131
```

The output of this command is the AWS AMI ID for your designated architecture and the **us-west-1** region. The AMI must belong to the same region as the cluster.

6.15.13. RHCOS AMIs for the AWS infrastructure

Red Hat provides Red Hat Enterprise Linux CoreOS (RHCOS) AMIs that are valid for the various AWS regions and instance architectures that you can manually specify for your OpenShift Container Platform nodes.



NOTE

By importing your own AMI, you can also install to regions that do not have a published RHCOS AMI.

Table 6.53. x86_64 RHCOS AMIs

AWS zone	AWS AMI
af-south-1	ami-073850a7021953a5c
ap-east-1	ami-0f8800a05c09be42d
ap-northeast-1	ami-0a226dbcc9a561c40
ap-northeast-2	ami-041ae0537e2eddec1
ap-northeast-3	ami-0bb8d9b69dc5b7670
ap-south-1	ami-0e9c18058fc5f94fd
ap-southeast-1	ami-03022d358ba2168be
ap-southeast-2	ami-09ffdc5be9b973be0
ap-southeast-3	ami-0facf1a0edeb20314
ca-central-1	ami-028cea206c2d03317
eu-central-1	ami-002eb441f329ccb0f
eu-north-1	ami-0b1a1fb68b3b9fee7
eu-south-1	ami-0bd0fd41a1d3f799a
eu-west-1	ami-04504e8799057980c

AWS zone	AWS AMI
eu-west-2	ami-0cc9297ddb3bce971
eu-west-3	ami-06f98f607a50937c6
me-south-1	ami-0fe39da7871a5b2a5
sa-east-1	ami-08265cc3226697767
us-east-1	ami-0fe05b1aa8dacfa90
us-east-2	ami-0ff64f495c7e977cf
us-gov-east-1	ami-0c99658076c41872a
us-gov-west-1	ami-0ca4acd5b8ba1cb1d
us-west-1	ami-01dc5d8e6bb6f23f4
us-west-2	ami-0404a109adfd00019

Table 6.54. aarch64 RHCOS AMIs

AWS zone	AWS AMI
af-south-1	ami-0574bcc5f80b0ad9a
ap-east-1	ami-0a65e79822ae2d235
ap-northeast-1	ami-0f7ef19d48e22353b
ap-northeast-2	ami-051dc6de359975e3c
ap-northeast-3	ami-0fd0b4222595650ac
ap-south-1	ami-05f9d14fe4a90ed6f
ap-southeast-1	ami-0afdb9133d22fba5f
ap-southeast-2	ami-0ef979abe82d07d44
ap-southeast-3	ami-025f9103ac4310e7f
ca-central-1	ami-0588cdf59e5c14847

AWS zone	AWS AMI
eu-central-1	ami-0ef24c0e18f93fa42
eu-north-1	ami-0439e2a3bf315df1a
eu-south-1	ami-0714e7c2e0106cdd3
eu-west-1	ami-0b960e76764ccd0c3
eu-west-2	ami-02621f50de62b3b89
eu-west-3	ami-0933ce7f5e2bfb50e
me-south-1	ami-074bde61a2ab740ee
sa-east-1	ami-03b4f97cfc8033ae0
us-east-1	ami-02a574449d4f4d280
us-east-2	ami-020e5600ef28c60ae
us-gov-east-1	ami-069f60e1dcf766d24
us-gov-west-1	ami-0db3cda4dbaccda02
us-west-1	ami-0c90cabeb5dee3178
us-west-2	ami-0f96437a23aeae53f

6.15.14. Creating the bootstrap node in AWS

You must create the bootstrap node in Amazon Web Services (AWS) to use during OpenShift Container Platform cluster initialization. You do this by:

- Providing a location to serve the **bootstrap.ign** Ignition config file to your cluster. This file is located in your installation directory. The provided CloudFormation Template assumes that the Ignition config files for your cluster are served from an S3 bucket. If you choose to serve the files from another location, you must modify the templates.
- Using the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the bootstrap node that your OpenShift Container Platform installation requires.



NOTE

If you do not use the provided CloudFormation template to create your bootstrap node, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.

Procedure

1. Create the bucket by running the following command:

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

- 1 **<cluster-name>-infra** is the bucket name. When creating the **install-config.yaml** file, replace **<cluster-name>** with the name specified for the cluster.

You must use a presigned URL for your S3 bucket, instead of the **s3://** schema, if you are:

- Deploying to a region that has endpoints that differ from the AWS SDK.
 - Deploying a proxy.
 - Providing your own custom endpoints.
2. Upload the **bootstrap.ign** Ignition config file to the bucket by running the following command:

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

3. Verify that the file uploaded by running the following command:

```
$ aws s3 ls s3://<cluster-name>-infra/
```

Example output

```
2019-04-03 16:15:16 314878 bootstrap.ign
```



NOTE

The bootstrap Ignition config file does contain secrets, like X.509 keys. The following steps provide basic security for the S3 bucket. To provide additional security, you can enable an S3 bucket policy to allow only certain users, such as the OpenShift IAM user, to access objects that the bucket contains. You can avoid S3 entirely and serve your bootstrap Ignition config file from any address that the bootstrap machine can reach.

4. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcocAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupid", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcId", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  }
]
```

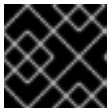
```

    },
    {
      "ParameterKey": "InternalApiTargetGroupArn", 21
      "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
    },
    {
      "ParameterKey": "InternalServiceTargetGroupArn", 23
      "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
    }
  ]

```

- 1** The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2** Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3** Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the bootstrap node based on your selected architecture.
- 4** Specify a valid **AWS::EC2::Image::Id** value.
- 5** CIDR block to allow SSH access to the bootstrap node.
- 6** Specify a CIDR block in the format **x.x.x.x/16-24**.
- 7** The public subnet that is associated with your VPC to launch the bootstrap node into.
- 8** Specify the **PublicSubnetIds** value from the output of the CloudFormation template for the VPC.
- 9** The master security group ID (for registering temporary rules)
- 10** Specify the **MasterSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
- 11** The VPC created resources will belong to.
- 12** Specify the **VpcId** value from the output of the CloudFormation template for the VPC.
- 13** Location to fetch bootstrap Ignition config file from.
- 14** Specify the S3 bucket and file name in the form **s3://<bucket_name>/bootstrap.ign**.
- 15** Whether or not to register a network load balancer (NLB).
- 16** Specify **yes** or **no**. If you specify **yes**, you must provide a Lambda Amazon Resource Name (ARN) value.
- 17** The ARN for NLB IP target registration lambda group.
- 18** Specify the **RegisterNlbTargetsLambda** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.

- 19 The ARN for external API load balancer target group.
 - 20 Specify the **ExternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 21 The ARN for internal API load balancer target group.
 - 22 Specify the **InternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 23 The ARN for internal service load balancer target group.
 - 24 Specify the **InternalServiceTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
5. Copy the template from the **CloudFormation template for the bootstrap machines** section of this topic and save it as a YAML file on your computer. This template describes the bootstrap machine that your cluster requires.
 6. Optional: If you are deploying the cluster with a proxy, you must update the ignition in the template to add the **ignition.config.proxy** fields. Additionally, if you have added the Amazon EC2, Elastic Load Balancing, and S3 VPC endpoints to your VPC, you must add these endpoints to the **noProxy** field.
 7. Launch the CloudFormation template to create a stack of AWS resources that represent the bootstrap node:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-bootstrap**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.
- 4 You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** and **AWS::IAM::InstanceProfile** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

- Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

Bootstrap InstanceId	The bootstrap Instance ID.
Bootstrap PublicIp	The bootstrap node public IP address.
Bootstrap PrivateIp	The bootstrap node private IP address.

6.15.14.1. CloudFormation template for the bootstrap machine

You can use the following CloudFormation template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster.

Example 6.71. CloudFormation template for the bootstrap machine

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\_]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^((([0-9]{1,3}[0-9]{1,3}[0-9]{1,3}[0-9]{1,3})|([0-9]{1,3}[0-9]{1,3}[0-9]{1,3})|([0-9]{1,3}[0-9]{1,3})|([0-9]{1,3})|([0-9]{1,3}))\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3})/([0-9]{1,2}|[0-9]{1,3})$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
```

```

    Type: AWS::EC2::Subnet::Id
MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
RegisterNlbTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String
BootstrapInstanceType:
    Description: Instance type for the bootstrap EC2 instance
    Default: "i3.large"
    Type: String

Metadata:
AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - PublicSubnet
    - Label:
        default: "Load Balancer Automation"

```

Parameters:

- AutoRegisterELB
- RegisterNLbIpTargetsLambdaArn
- ExternalApiTargetGroupArn
- InternalApiTargetGroupArn
- InternalServiceTargetGroupArn

ParameterLabels:

InfrastructureName:

default: "Infrastructure Name"

VpcId:

default: "VPC ID"

AllowedBootstrapSshCidr:

default: "Allowed SSH Source"

PublicSubnet:

default: "Public Subnet"

RhcosAmi:

default: "Red Hat Enterprise Linux CoreOS AMI ID"

BootstrapIgnitionLocation:

default: "Bootstrap Ignition Source"

MasterSecurityGroupId:

default: "Master Security Group ID"

AutoRegisterELB:

default: "Use Provided ELB Automation"

Conditions:

DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:

BootstrapIamRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Principal:

Service:

- "ec2.amazonaws.com"

Action:

- "sts:AssumeRole"

Path: "/"

Policies:

- PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: "Allow"

Action: "ec2:Describe*"

Resource: "*"

- Effect: "Allow"

Action: "ec2:AttachVolume"

Resource: "*"

- Effect: "Allow"

Action: "ec2:DetachVolume"

Resource: "*"

- Effect: "Allow"

Action: "s3:GetObject"

Resource: "*"

BootstrapInstanceProfile:

Type: "AWS::IAM::InstanceProfile"

Properties:

Path: "/"

Roles:

- Ref: "BootstrapIamRole"

BootstrapSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Cluster Bootstrap Security Group

SecurityGroupIngress:

- IpProtocol: tcp

FromPort: 22

ToPort: 22

CidrIp: !Ref AllowedBootstrapSshCidr

- IpProtocol: tcp

ToPort: 19531

FromPort: 19531

CidrIp: 0.0.0.0/0

VpcId: !Ref VpcId

BootstrapInstance:

Type: AWS::EC2::Instance

Properties:

ImageId: !Ref RhcosAmi

IamInstanceProfile: !Ref BootstrapInstanceProfile

InstanceType: !Ref BootstrapInstanceType

NetworkInterfaces:

- AssociatePublicIpAddress: "true"

DeviceIndex: "0"

GroupSet:

- !Ref "BootstrapSecurityGroup"

- !Ref "MasterSecurityGroupId"

SubnetId: !Ref "PublicSubnet"

UserData:

Fn::Base64: !Sub

- '{"ignition":{"config":{"replace":{"source":"\${S3Loc}"},"version":"3.1.0"}}}'

- {

 S3Loc: !Ref BootstrapIgnitionLocation

}

RegisterBootstrapApiTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:

ServiceToken: !Ref RegisterNlbTargetsLambdaArn

TargetArn: !Ref ExternalApiTargetGroupArn

TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalApiTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:
 ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
 TargetArn: !Ref InternalApiTargetGroupArn
 TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalServiceTarget:
 Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
 TargetArn: !Ref InternalServiceTargetGroupArn
 TargetIp: !GetAtt BootstrapInstance.PrivateIp

Outputs:
 BootstrapInstanceId:
 Description: Bootstrap Instance ID.
 Value: !Ref BootstrapInstance

BootstrapPublicIp:
 Description: The bootstrap node public IP address.
 Value: !GetAtt BootstrapInstance.PublicIp

BootstrapPrivateIp:
 Description: The bootstrap node private IP address.
 Value: !GetAtt BootstrapInstance.PrivateIp

Additional resources

- See [RHCOS AMIs for the AWS infrastructure](#) for details about the Red Hat Enterprise Linux CoreOS (RHCOS) AMIs for the AWS zones.

6.15.15. Creating the control plane machines in AWS

You must create the control plane machines in Amazon Web Services (AWS) that your cluster will use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the control plane nodes.



IMPORTANT

The CloudFormation template creates a stack that represents three control plane nodes.



NOTE

If you do not use the provided CloudFormation template to create your control plane nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.

- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.

Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcOsAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
  {
    "ParameterKey": "Master0Subnet", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "Master1Subnet", 13
    "ParameterValue": "subnet-<random_string>" 14
  },
  {
    "ParameterKey": "Master2Subnet", 15
    "ParameterValue": "subnet-<random_string>" 16
  },
  {
    "ParameterKey": "MasterSecurityGroupID", 17
    "ParameterValue": "sg-<random_string>" 18
  },
  {

```

```

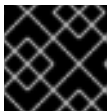
    "ParameterKey": "IgnitionLocation", 19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
  },
  {
    "ParameterKey": "CertificateAuthorities", 21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" 22
  },
  {
    "ParameterKey": "MasterInstanceProfileName", 23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
  },
  {
    "ParameterKey": "MasterInstanceType", 25
    "ParameterValue": "" 26
  },
  {
    "ParameterKey": "AutoRegisterELB", 27
    "ParameterValue": "yes" 28
  },
  {
    "ParameterKey": "RegisterNlbTargetsLambdaArn", 29
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<random_string>" 30
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]

```

- 1 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3 Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the control plane machines based on your selected architecture.
- 4 Specify an **AWS::EC2::Image::Id** value.
- 5 Whether or not to perform DNS etcd registration.

- 6 Specify **yes** or **no**. If you specify **yes**, you must provide hosted zone information.
- 7 The Route 53 private zone ID to register the etcd targets with.
- 8 Specify the **PrivateHostedZoneId** value from the output of the CloudFormation template for DNS and load balancing.
- 9 The Route 53 zone to register the targets with.
- 10 Specify **<cluster_name>.<domain_name>** where **<domain_name>** is the Route 53 base domain that you used when you generated **install-config.yaml** file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
- 11 13 15 A subnet, preferably private, to launch the control plane machines on.
- 12 14 16 Specify a subnet from the **PrivateSubnets** value from the output of the CloudFormation template for DNS and load balancing.
- 17 The master security group ID to associate with control plane nodes.
- 18 Specify the **MasterSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
- 19 The location to fetch control plane Ignition config file from.
- 20 Specify the generated Ignition config file location, https://api-int.<cluster_name>.<domain_name>:22623/config/master.
- 21 The base64 encoded certificate authority string to use.
- 22 Specify the value from the **master.ign** file that is in the installation directory. This value is the long string with the format **data:text/plain;charset=utf-8;base64,ABC...xYz==**.
- 23 The IAM profile to associate with control plane nodes.
- 24 Specify the **MasterInstanceProfile** parameter value from the output of the CloudFormation template for the security group and roles.
- 25 The type of AWS instance to use for the control plane machines based on your selected architecture.
- 26 The instance type value corresponds to the minimum resource requirements for control plane machines. For example **m6i.xlarge** is a type for AMD64, and **m6g.xlarge** is a type for ARM64.
- 27 Whether or not to register a network load balancer (NLB).
- 28 Specify **yes** or **no**. If you specify **yes**, you must provide a Lambda Amazon Resource Name (ARN) value.
- 29 The ARN for NLB IP target registration lambda group.
- 30 Specify the **RegisterNlbTargetsLambda** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
- 31 The ARN for external API load balancer target group.

- 32 Specify the **ExternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an
 - 33 The ARN for internal API load balancer target group.
 - 34 Specify the **InternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
 - 35 The ARN for internal service load balancer target group.
 - 36 Specify the **InternalServiceTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use **arn:aws-us-gov** if deploying the cluster to an AWS GovCloud region.
2. Copy the template from the **CloudFormation template for control plane machines** section of this topic and save it as a YAML file on your computer. This template describes the control plane machines that your cluster requires.
 3. If you specified an **m5** instance type as the value for **MasterInstanceType**, add that instance type to the **MasterInstanceType.AllowedValues** parameter in the CloudFormation template.
 4. Launch the CloudFormation template to create a stack of AWS resources that represent the control plane nodes:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
```

- 1 **<name>** is the name for the CloudFormation stack, such as **cluster-control-plane**. You need the name of this stack if you remove the cluster.
- 2 **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3 **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```



NOTE

The CloudFormation template creates a stack that represents three control plane nodes.

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6.15.15.1. CloudFormation template for control plane machines

You can use the following CloudFormation template to deploy the control plane machines that you need for your OpenShift Container Platform cluster.

Example 6.72. CloudFormation template for control plane machines

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\_]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneId:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneName:
    Default: ""
    Description: unused
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
```

Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
 Description: Base64 encoded certificate authority string to use.
 Type: String

MasterInstanceProfileName:

Description: IAM profile to associate with master nodes.
 Type: String

MasterInstanceType:

Default: m5.xlarge
 Type: String

AutoRegisterELB:

Default: "yes"

AllowedValues:

- "yes"
- "no"

Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
 Type: String

RegisterNlbTargetsLambdaArn:

Description: ARN for NLB IP target registration lambda. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

ExternalApiTargetGroupArn:

Description: ARN for external API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

InternalApiTargetGroupArn:

Description: ARN for internal API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

InternalServiceTargetGroupArn:

Description: ARN for internal service load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.

Type: String

Metadata:

AWS::CloudFormation::Interface:

ParameterGroups:

- Label:

default: "Cluster Information"

Parameters:

- InfrastructureName

- Label:

default: "Host Information"

Parameters:

- MasterInstanceType

- RhcosAmi

- IgnitionLocation

- CertificateAuthorities

- MasterSecurityGroupId

- MasterInstanceProfileName

- Label:

default: "Network Configuration"

Parameters:

- VpcId

- AllowedBootstrapSshCidr

- Master0Subnet

- Master1Subnet
- Master2Subnet
- Label:
 - default: "Load Balancer Automation"
- Parameters:
 - AutoRegisterELB
 - RegisterNlbIpTargetsLambdaArn
 - ExternalApiTargetGroupArn
 - InternalApiTargetGroupArn
 - InternalServiceTargetGroupArn
- ParameterLabels:
 - InfrastructureName:
 - default: "Infrastructure Name"
 - VpcId:
 - default: "VPC ID"
 - Master0Subnet:
 - default: "Master-0 Subnet"
 - Master1Subnet:
 - default: "Master-1 Subnet"
 - Master2Subnet:
 - default: "Master-2 Subnet"
 - MasterInstanceType:
 - default: "Master Instance Type"
 - MasterInstanceProfileName:
 - default: "Master Instance Profile Name"
 - RhcosAmi:
 - default: "Red Hat Enterprise Linux CoreOS AMI ID"
 - BootstrapIgnitionLocation:
 - default: "Master Ignition Source"
 - CertificateAuthorities:
 - default: "Ignition CA String"
 - MasterSecurityGroupId:
 - default: "Master Security Group ID"
 - AutoRegisterELB:
 - default: "Use Provided ELB Automation"

Conditions:

DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:

Master0:

Type: AWS::EC2::Instance

Properties:

ImageId: !Ref RhcosAmi

BlockDeviceMappings:

- DeviceName: /dev/xvda

Ebs:

- VolumeSize: "120"

- VolumeType: "gp2"

IamInstanceProfile: !Ref MasterInstanceProfileName

InstanceType: !Ref MasterInstanceType

NetworkInterfaces:

- AssociatePublicIpAddress: "false"

- DeviceIndex: "0"

GroupSet:

- !Ref "MasterSecurityGroupId"

```

    SubnetId: !Ref "Master0Subnet"
  UserData:
    Fn::Base64: !Sub
      - '{"ignition":{"config":{"merge":{"source":"${SOURCE}"}}, "security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}}, "version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
  Tags:
    - Key: !Join [ "", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

```

```

RegisterMaster0:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

```

```

RegisterMaster0InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

```

```

RegisterMaster0InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

```

```

Master1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master1Subnet"
  UserData:
    Fn::Base64: !Sub

```

```

- {"ignition":{"config":{"merge":{"source":"${SOURCE}"},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"]}},"version":"3.1.0"}}
- {
  SOURCE: !Ref IgnitionLocation,
  CA_BUNDLE: !Ref CertificateAuthorities,
}
Tags:
- Key: !Join [ "", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
  Value: "shared"

```

RegisterMaster1:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

```

RegisterMaster1InternalApiTarget:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbTargetsLambdaArn
  TargetArn: !Ref InternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

```

RegisterMaster1InternalServiceTarget:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbTargetsLambdaArn
  TargetArn: !Ref InternalServiceTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

```

Master2:

```

Type: AWS::EC2::Instance
Properties:
  ImageId: !Ref RhcosAmi
  BlockDeviceMappings:
  - DeviceName: /dev/xvda
    Ebs:
      VolumeSize: "120"
      VolumeType: "gp2"
  IamInstanceProfile: !Ref MasterInstanceProfileName
  InstanceType: !Ref MasterInstanceType
  NetworkInterfaces:
  - AssociatePublicIpAddress: "false"
    DeviceIndex: "0"
    GroupSet:
    - !Ref "MasterSecurityGroupId"
    SubnetId: !Ref "Master2Subnet"
  UserData:
    Fn::Base64: !Sub
      - {"ignition":{"config":{"merge":{"source":"${SOURCE}"},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"]}},"version":"3.1.0"}}
      - {

```

```

SOURCE: !Ref IgnitionLocation,
CA_BUNDLE: !Ref CertificateAuthorities,
}
Tags:
- Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
  Value: "shared"

```

RegisterMaster2:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

```

RegisterMaster2InternalApiTarget:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalApiTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

```

RegisterMaster2InternalServiceTarget:

```

Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalServiceTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

```

Outputs:**PrivateIPs:**

Description: The control-plane node private IP addresses.

Value:

```

!Join [
  "",
  [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
]

```

6.15.16. Creating the worker nodes in AWS

You can create worker nodes in Amazon Web Services (AWS) for your cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent a worker node.

**IMPORTANT**

The CloudFormation template creates a stack that represents one worker node. You must create a stack for each worker node.



NOTE

If you do not use the provided CloudFormation template to create your worker nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.

Procedure

1. Create a JSON file that contains the parameter values that the CloudFormation template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcocAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupld", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  }
]
```

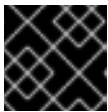
```

    },
    {
      "ParameterKey": "WorkerInstanceProfileName", 13
      "ParameterValue": "" 14
    },
    {
      "ParameterKey": "WorkerInstanceType", 15
      "ParameterValue": "" 16
    }
  ]

```

- 1 The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
- 2 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format **<cluster-name>-<random-string>**.
- 3 Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the worker nodes based on your selected architecture.
- 4 Specify an **AWS::EC2::Image::Id** value.
- 5 A subnet, preferably private, to start the worker nodes on.
- 6 Specify a subnet from the **PrivateSubnets** value from the output of the CloudFormation template for DNS and load balancing.
- 7 The worker security group ID to associate with worker nodes.
- 8 Specify the **WorkerSecurityGroupId** value from the output of the CloudFormation template for the security group and roles.
- 9 The location to fetch the bootstrap Ignition config file from.
- 10 Specify the generated Ignition config location, https://api-int.<cluster_name>.<domain_name>:22623/config/worker.
- 11 Base64 encoded certificate authority string to use.
- 12 Specify the value from the **worker.ign** file that is in the installation directory. This value is the long string with the format **data:text/plain;charset=utf-8;base64,ABC...xYz==**.
- 13 The IAM profile to associate with worker nodes.
- 14 Specify the **WorkerInstanceProfile** parameter value from the output of the CloudFormation template for the security group and roles.
- 15 The type of AWS instance to use for the compute machines based on your selected architecture.
- 16 The instance type value corresponds to the minimum resource requirements for compute machines. For example **m6i.large** is a type for AMD64. and **m6g.large** is a type for ARM64.

2. Copy the template from the **CloudFormation template for worker machines** section of this topic and save it as a YAML file on your computer. This template describes the networking objects and load balancers that your cluster requires.
3. Optional: If you specified an **m5** instance type as the value for **WorkerInstanceType**, add that instance type to the **WorkerInstanceType.AllowedValues** parameter in the CloudFormation template.
4. Optional: If you are deploying with an AWS Marketplace image, update the **Worker0.type.properties.ImageID** parameter with the AMI ID that you obtained from your subscription.
5. Use the CloudFormation template to create a stack of AWS resources that represent a worker node:



IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
  --template-body file://<template>.yaml \ 2
  --parameters file://<parameters>.json 3
```

- 1** **<name>** is the name for the CloudFormation stack, such as **cluster-worker-1**. You need the name of this stack if you remove the cluster.
- 2** **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.
- 3** **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-11eb-348f-sd9888c65b59
```



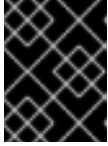
NOTE

The CloudFormation template creates a stack that represents one worker node.

6. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

7. Continue to create worker stacks until you have created enough worker machines for your cluster. You can create additional worker stacks by referencing the same template and parameter files and specifying a different stack name.

**IMPORTANT**

You must create at least two worker machines, so you must create at least two stacks that use this CloudFormation template.

6.15.16.1. CloudFormation template for worker machines

You can use the following CloudFormation template to deploy the worker machines that you need for your OpenShift Container Platform cluster.

Example 6.73. CloudFormation template for worker machines

```

AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-\]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  WorkerInstanceType:
    Default: m5.large
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Cluster Information"
        Parameters:
          - InfrastructureName
          - Label:

```



```

    default: "Host Information"
Parameters:
- WorkerInstanceType
- RhcosAmi
- IgnitionLocation
- CertificateAuthorities
- WorkerSecurityGroupld
- WorkerInstanceProfileName
- Label:
  default: "Network Configuration"
Parameters:
- Subnet
ParameterLabels:
Subnet:
  default: "Subnet"
InfrastructureName:
  default: "Infrastructure Name"
WorkerInstanceType:
  default: "Worker Instance Type"
WorkerInstanceProfileName:
  default: "Worker Instance Profile Name"
RhcosAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"
IgnitionLocation:
  default: "Worker Ignition Source"
CertificateAuthorities:
  default: "Ignition CA String"
WorkerSecurityGroupld:
  default: "Worker Security Group ID"

Resources:
Worker0:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
    IamInstanceProfile: !Ref WorkerInstanceProfileName
    InstanceType: !Ref WorkerInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "WorkerSecurityGroupld"
        SubnetId: !Ref "Subnet"
    UserData:
      Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }

```

Tags:

- Key: !Join [""], ["kubernetes.io/cluster/", !Ref InfrastructureName]]
- Value: "shared"

Outputs:**PrivateIP:**

- Description: The compute node private IP address.
- Value: !GetAtt Worker0.PrivateIp

6.15.17. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure

After you create all of the required infrastructure in Amazon Web Services (AWS), you can start the bootstrap sequence that initializes the OpenShift Container Platform control plane.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running **aws configure**.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.
- You created the worker nodes.

Procedure

1. Change to the directory that contains the installation program and start the bootstrap process that initializes the OpenShift Container Platform control plane:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 20m0s for the Kubernetes API at
https://api.mycluster.example.com:6443...
```

```
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```

If the command exits without a **FATAL** warning, your OpenShift Container Platform control plane has initialized.



NOTE

After the control plane initializes, it sets up the compute nodes and installs additional services in the form of Operators.

Additional resources

- See [Monitoring installation progress](#) for details about monitoring the installation, bootstrap, and control plane logs as an OpenShift Container Platform installation progresses.
- See [Gathering bootstrap node diagnostic data](#) for information about troubleshooting issues related to the bootstrap process.

6.15.18. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.15.19. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

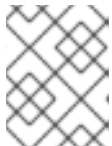
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

6.15.20. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

- Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

6.15.20.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

6.15.20.2. Image registry storage configuration

Amazon Web Services provides default storage, which means the Image Registry Operator is available after installation. However, if the Registry Operator cannot create an S3 bucket and automatically configure storage, you must manually configure registry storage.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

6.15.20.2.1. Configuring registry storage for AWS with user-provisioned infrastructure

During installation, your cloud credentials are sufficient to create an Amazon S3 bucket and the Registry Operator will automatically configure storage.

If the Registry Operator cannot create an S3 bucket and automatically configure storage, you can create an S3 bucket and configure storage with the following procedure.

Prerequisites

- You have a cluster on AWS with user-provisioned infrastructure.
- For Amazon S3 storage, the secret is expected to contain two keys:
 - **REGISTRY_STORAGE_S3_ACCESSKEY**
 - **REGISTRY_STORAGE_S3_SECRETKEY**

Procedure

Use the following procedure if the Registry Operator cannot create an S3 bucket and automatically configure storage.

1. Set up a [Bucket Lifecycle Policy](#) to abort incomplete multipart uploads that are one day old.
2. Fill in the storage configuration in **configs.imageregistry.operator.openshift.io/cluster**:

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

Example configuration

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```


**WARNING**

To secure your registry images in AWS, [block public access](#) to the S3 bucket.

6.15.20.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

6.15.21. Deleting the bootstrap resources

After you complete the initial Operator configuration for the cluster, remove the bootstrap resources from Amazon Web Services (AWS).

Prerequisites

- You completed the initial Operator configuration for your cluster.

Procedure

- Delete the bootstrap resources. If you used the CloudFormation template, [delete its stack](#):
 - Delete the stack by using the AWS CLI:

```
$ aws cloudformation delete-stack --stack-name <name> 1
```

1 `<name>` is the name of your bootstrap stack.

- Delete the stack by using the [AWS CloudFormation console](#).

6.15.22. Creating the Ingress DNS Records

If you removed the DNS Zone configuration, manually create DNS records that point to the Ingress load balancer. You can create either a wildcard record or specific records. While the following procedure uses A records, you can use other record types that you require, such as CNAME or alias.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Amazon Web Services (AWS) that uses infrastructure that you provisioned.
- You installed the OpenShift CLI (**oc**).
- You installed the **jq** package.
- You downloaded the AWS CLI and installed it on your computer. See [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#).

Procedure

1. Determine the routes to create.
 - To create a wildcard record, use ***.apps.<cluster_name>.<domain_name>**, where **<cluster_name>** is your cluster name, and **<domain_name>** is the Route 53 base domain for your OpenShift Container Platform cluster.
 - To create specific records, you must create a record for each route that your cluster uses, as shown in the output of the following command:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

2. Retrieve the Ingress Operator load balancer status and note the value of the external IP address that it uses, which is shown in the **EXTERNAL-IP** column:

```
$ oc -n openshift-ingress get service router-default
```

Example output

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
AGE				

```
router-default LoadBalancer 172.30.62.215 ab3...28.us-east-2.elb.amazonaws.com
80:31499/TCP,443:30693/TCP 5m
```

3. Locate the hosted zone ID for the load balancer:

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
"<external_ip>").CanonicalHostedZoneNameID' 1
```

- 1** For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer that you obtained.

Example output

```
Z3AADJGX6KTTL2
```

The output of this command is the load balancer hosted zone ID.

4. Obtain the public hosted zone ID for your cluster's domain:

```
$ aws route53 list-hosted-zones-by-name \
  --dns-name "<domain_name>" \ 1
  --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2
  --output text
```

- 1** **2** For **<domain_name>**, specify the Route 53 base domain for your OpenShift Container Platform cluster.

Example output

```
/hostedzone/Z3URY6TWQ91KVV
```

The public hosted zone ID for your domain is shown in the command output. In this example, it is **Z3URY6TWQ91KVV**.

5. Add the alias records to your private zone:

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ 1
> "Changes": [
> {
>   "Action": "CREATE",
>   "ResourceRecordSet": {
>     "Name": "\\052.apps.<cluster_domain>", 2
>     "Type": "A",
>     "AliasTarget":{
>       "HostedZoneId": "<hosted_zone_id>", 3
>       "DNSName": "<external_ip>.", 4
>       "EvaluateTargetHealth": false
>     }
>   }
> }
```

```
> }
> ]
> }'
```

- 1 For **<private_hosted_zone_id>**, specify the value from the output of the CloudFormation template for DNS and load balancing.
- 2 For **<cluster_domain>**, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.
- 3 For **<hosted_zone_id>**, specify the public hosted zone ID for the load balancer that you obtained.
- 4 For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6. Add the records to your public zone:

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --
change-batch '{ 1
> "Changes": [
> {
>   "Action": "CREATE",
>   "ResourceRecordSet": {
>     "Name": "\\052.apps.<cluster_domain>", 2
>     "Type": "A",
>     "AliasTarget":{
>       "HostedZoneId": "<hosted_zone_id>", 3
>       "DNSName": "<external_ip>.", 4
>       "EvaluateTargetHealth": false
>     }
>   }
> }
> ]
> }'
```

- 1 For **<public_hosted_zone_id>**, specify the public hosted zone for your domain.
- 2 For **<cluster_domain>**, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.
- 3 For **<hosted_zone_id>**, specify the public hosted zone ID for the load balancer that you obtained.
- 4 For **<external_ip>**, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6.15.23. Completing an AWS installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Amazon Web Service (AWS) user-provisioned infrastructure, monitor the deployment to completion.

Prerequisites

- You removed the bootstrap node for an OpenShift Container Platform cluster on user-provisioned AWS infrastructure.
- You installed the **oc** CLI.

Procedure

1. From the directory that contains the installation program, complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 1s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrap** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Register your cluster on the [Cluster registration](#) page.

6.15.24. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

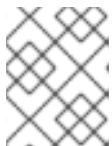
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

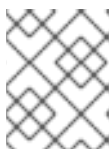


NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

6.15.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

6.15.26. Additional resources

- See [Working with stacks](#) in the AWS documentation for more information about AWS CloudFormation stacks.

6.15.27. Next steps

- [Validate an installation.](#)
- [Customize your cluster.](#)
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#) .
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)
- If necessary, you can [remove cloud provider credentials](#).

6.16. INSTALLING A CLUSTER ON AWS WITH REMOTE WORKERS ON AWS OUTPOSTS

In OpenShift Container Platform version 4.12, you can install a cluster on Amazon Web Services (AWS) with remote workers running in AWS Outposts. This can be achieved by customizing the default AWS installation and performing some manual steps.

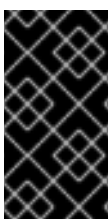


IMPORTANT

Installing a cluster on AWS with remote workers on AWS Outposts is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

For more info about AWS Outposts see [AWS Outposts Documentation](#) .



IMPORTANT

In order to install a cluster with remote workers in AWS Outposts, all worker instances must be located within the same Outpost instance and cannot be located in an AWS region. It is not possible for the cluster to have instances in both AWS Outposts and AWS region. In addition, it also follows that control plane nodes mustn't be schedulable.

6.16.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update processes](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an AWS account](#) to host the cluster.
- You are familiar with the instance types are supported in the AWS Outpost instance you use. This can be validated with [get-outpost-instance-types AWS CLI command](#)
- You are familiar with the AWS Outpost instance details, such as OutpostArn and AvailabilityZone. This can be validated with [list-outposts AWS CLI command](#)



IMPORTANT

Since the cluster uses the provided AWS credentials to create AWS resources for its entire life cycle, the credentials must be key-based and long-lived. So, If you have an AWS profile stored on your computer, it must not use a temporary session token, generated while using a multi-factor authentication device. For more information about generating the appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You may supply the keys when you run the installation program.

- You have access to an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). See the section "About using a custom VPC" for more information.
- If a firewall is used, it was [configured to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

6.16.2. About using a custom VPC

OpenShift Container Platform 4.12 installer cannot automatically deploy AWS Subnets on AWS Outposts, so you will need to manually configure the VPC. Therefore, you have to deploy the cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). In addition, by deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

6.16.2.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways

- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See [Amazon VPC console wizard configurations](#) and [Work with VPCs and subnets](#) in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See [VPC networking components](#) and [Route tables for your VPC](#) for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:



NOTE

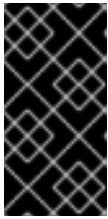
To allow the creation of OpenShift Container Platform with remote workers in the AWS Outposts, you must create at least one private subnet in the AWS Outpost instance for the workload instances creation and one private subnet in an AWS region for the control plane instances creation. If you specify more than one private subnet in the region, the control plane instances will be distributed across these subnets. You will also need to create a public subnet in each of the availability zones used for private subnets, including the Outpost private subnet, as Network Load Balancers will be created in the AWS region for the API server and Ingress network as part of the cluster installation. It is possible to create an AWS region private subnet in the same Availability zone as an Outpost private subnet.

- Create a public and private subnet in the AWS Region for each availability zone that your control plane uses. Each availability zone can contain no more than one public and one private subnet in the AWS region. For an example of this type of configuration, see [VPC with public and private subnets \(NAT\)](#) in the AWS documentation.

To create a private subnet in the AWS Outposts, you need to first ensure that the Outpost instance is located in the desired availability zone. Then, you can create the private subnet within that availability zone within the Outpost instance, by adding the Outpost ARN. Make sure there is another public subnet in the AWS Region created in the same availability zone.

Record each subnet ID. Completing the installation requires that you enter all the subnets IDs,

created in the AWS Region, in the **platform** section of the **install-config.yaml** file and changing the workers **machineset** to use the private subnet ID created in the Outpost. See [Finding a subnet ID](#) in the AWS documentation.



IMPORTANT

In case you need to create a public subnet in the AWS Outposts, verify that this subnet is not used for the Network or Classic LoadBalancer, otherwise the LoadBalancer creation fails. To achieve that, the **kubernetes.io/cluster/*-outposts: owned** special tag must be included in the subnet.

- The VPC's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines. The subnet CIDR blocks must belong to the machine CIDR that you specify.
- The VPC must have a public internet gateway attached to it. For each availability zone:
 - The public subnet requires a route to the internet gateway.
 - The public subnet requires a NAT gateway with an EIP address.
 - The private subnet requires a route to the NAT gateway in public subnet.



NOTE

To access your local cluster over your local network, the VPC must be associated with your Outpost's local gateway route table. For more information, see [VPC associations](#) in the AWS Outposts User Guide.

- The VPC must not use the **kubernetes.io/cluster/*: owned, Name**, and **openshift.io/cluster** tags.
The installation program modifies your subnets to add the **kubernetes.io/cluster/*: shared** tag, so your subnets must have at least one free tag slot available for it. See [Tag Restrictions](#) in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify. You cannot use a **Name** tag, because it overlaps with the EC2 **Name** field and the installation fails.
- You must enable the **enableDnsSupport** and **enableDnsHostnames** attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster's internal DNS records. See [DNS Support in Your VPC](#) in the AWS documentation.
If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the **platform.aws.hostedZone** field in the **install-config.yaml** file.

Option 1: Create VPC endpoints

Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

With this option, network traffic remains private between your VPC and the required AWS services.

Option 2: Create a proxy without VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy. With this option, internet traffic goes through the proxy to reach the required AWS services.

Option 3: Create a proxy with VPC endpoints

As part of the installation process, you can configure an HTTP or HTTPS proxy with VPC endpoints. Create a VPC endpoint and attach it to the subnets that the clusters are using. Name the endpoints as follows:

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

When configuring the proxy in the **install-config.yaml** file, add these endpoints to the **noProxy** field. With this option, the proxy prevents the cluster from accessing the internet directly. However, network traffic remains private between your VPC and the required AWS services.

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

Component	AWS type	Description
VPC	<ul style="list-style-type: none"> • AWS::EC2::VPC • AWS::EC2::VPCEndpoint 	You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.
Public subnets	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkACLAssociation 	Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.
Internet gateway	<ul style="list-style-type: none"> • AWS::EC2::InternetGateway • AWS::EC2::VPCGatewayAttachment • AWS::EC2::RouteTable • AWS::EC2::Route • AWS::EC2::SubnetRouteTableAssociation • AWS::EC2::NatGateway • AWS::EC2::EIP 	You must have a public internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the internet and are not required for some restricted network or proxy scenarios.

Component	AWS type	Description	
Network access control	<ul style="list-style-type: none"> ● AWS::EC2::NetworkAcl ● AWS::EC2::NetworkAclEntry 	You must allow the VPC to access the following ports:	
		Port	Reason
		80	Inbound HTTP traffic
		443	Inbound HTTPS traffic
		22	Inbound SSH traffic
		1024 - 65535	Inbound ephemeral traffic
		0 - 65535	Outbound ephemeral traffic
Private subnets	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. To enable remote workers running in the Outpost, the VPC must include a private subnet located within the Outpost instance, in addition to the private subnets located within the corresponding AWS region. If you use private subnets, you must provide appropriate routes and tables for them.	

6.16.2.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains exactly one public and one private subnet in the AWS region (not created in the Outpost instance). The availability zone in which the Outpost instance is installed should include one additional private subnet in the Outpost instance.

- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the **kubernetes.io/cluster/.*: shared** tag is removed from the subnets that it used.

6.16.2.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

6.16.2.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

6.16.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

6.16.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

6.16.5. Obtaining the installation program

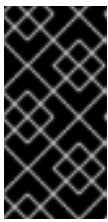
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

- Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
- Select your infrastructure provider.
- Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

- Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

6.16.6. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 6.55. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

6.16.7. Identifying your AWS Outposts instance types

AWS Outposts rack catalog includes options supporting the latest generation Intel powered EC2 instance types with or without local instance storage. Identify which instance types are configured in your AWS Outpost instance. As part of the installation process, you must update the **install-config.yaml** file with the instance type that the installation program will use to deploy worker nodes.

Procedure

Use the AWS CLI to get the list of supported instance types by running the following command:

```
$ aws outposts get-outpost-instance-types --outpost-id <outpost_id> 1
```

- 1 For **<outpost_id>**, specify the Outpost ID, used in the AWS account for the worker instances



IMPORTANT

When you purchase capacity for your AWS Outpost instance, you specify an EC2 capacity layout that each server provides. Each server supports a single family of instance types. A layout can offer a single instance type or multiple instance types. Dedicated Hosts allows you to alter whatever you chose for that initial layout. If you allocate a host to support a single instance type for the entire capacity, you can only start a single instance type from that host.

Supported instance types in AWS Outposts might be changed. For more information, you can check the [Compute and Storage](#) page in AWS Outposts documents.

6.16.8. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

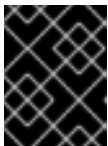
When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **AWS** as the platform to target.
 - iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.
 - iv. Select the AWS region to deploy the cluster to.
 - v. Select the base domain for the Route 53 service that you configured for your cluster.
 - vi. Enter a descriptive name for your cluster.
 - vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. The AWS Outposts installation has the following limitations which require manual modification of the **install-config.yaml** file:
- Unlike AWS Regions, which offer near-infinite scale, AWS Outposts are limited by their provisioned capacity, EC2 family and generations, configured instance sizes, and availability of compute capacity that is not already consumed by other workloads. Therefore, when creating new OpenShift Container Platform cluster, you need to provide the supported instance type in the **compute.platform.aws.type** section in the configuration file.
 - When deploying OpenShift Container Platform cluster with remote workers running in AWS Outposts, only one Availability Zone can be used for the compute instances - the Availability Zone in which the Outpost instance was created in. Therefore, when creating new OpenShift Container Platform cluster, it recommended to provide the relevant Availability Zone in the **compute.platform.aws.zones** section in the configuration file, in order to limit the compute instances to this Availability Zone.
 - Amazon Elastic Block Store (EBS) gp3 volumes aren't supported by the AWS Outposts service. This volume type is the default type used by the OpenShift Container Platform cluster. Therefore, when creating new OpenShift Container Platform cluster, you must change the volume type in the **compute.platform.aws.rootVolume.type** section to gp2. You will find more information about how to change these values below.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

6.16.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

6.16.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 6.56. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code> . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

6.16.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 6.57. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

6.16.8.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 6.58. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; font-weight: bold; text-align: center;">  </div> <p style="text-align: center;">IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="486 1440 592 1637" style="background-color: black; color: white; padding: 5px; font-weight: bold; text-align: center;">  </div> <p style="text-align: center;">NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true


Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
platform.aws.lbType	Required to set the NLB load balancer type in AWS. Valid values are Classic or NLB . If no value is specified, the installation program defaults to Classic . The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	Classic or NLB . The default value is Classic .
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

6.16.8.1.4. Optional AWS configuration parameters


Optional AWS configuration parameters are described in the following table:

Table 6.59. Optional AWS parameters

Parameter	Description	Values
compute.platform.aws.amiID	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
compute.platform.aws.iamRole	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
compute.platform.aws.rootVolume.iops	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example 4000 .
compute.platform.aws.rootVolume.size	The size in GiB of the root volume.	Integer, for example 500 .
compute.platform.aws.rootVolume.type	The type of the root volume.	Valid AWS EBS volume type , such as io1 .
compute.platform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN
compute.platform.aws.type	The EC2 instance type for the compute machines.	Valid AWS instance type, such as m4.2xlarge . See the Supported AWS machine types table that follows.
compute.platform.aws.zones	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .

Parameter	Description	Values
compute.aws.region	The AWS region that the installation program creates compute resources in.	<p>Any valid AWS region, such as us-east-1. You can use the AWS CLI to access the regions available based on your selected instance type. For example:</p> <pre>aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge</pre> <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>IMPORTANT</p> <p>When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.</p> </div> </div>
controlPlane.plattform.aws.amiID	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
controlPlane.plattform.aws.iamRole	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
controlPlane.plattform.aws.rootVolume.kmsKeyARN	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane.plattform.aws.type	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as m6i.xlarge . See the Supported AWS machine types table that follows.

Parameter	Description	Values
controlPlane.platform.aws.zones	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as us-east-1c , in a YAML sequence .
controlPlane.aws.region	The AWS region that the installation program creates control plane resources in.	Valid AWS region , such as us-east-1 .
platform.aws.amiID	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See <i>RHCOS AMIs for AWS infrastructure</i> for available AMI IDs.
platform.aws.hostedZone	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example Z3URY6TWQ91KVV .
platform.aws.serviceEndpoints.name	The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name.
platform.aws.serviceEndpoints.url	The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.	Valid AWS service endpoint URL.

Parameter	Description	Values
platform.aws.userTags	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  NOTE You can add up to 25 user defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform.aws.propagateUserTags	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example true or false .
platform.aws.subnets	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.	Valid subnet IDs.

6.16.8.2. Sample customized install-config.yaml file for AWS

You can customize the installation configuration file (**install-config.yaml**) to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
hyperthreading: Enabled 5

```

```

name: master
platform: {}
replicas: 3
compute: 6
- hyperthreading: Enabled 7
name: worker
platform:
  aws:
    type: m5.large 8
    zones:
      - us-east-1a 9
    rootVolume:
      type: gp2 10
      size: 120
    replicas: 3
  metadata:
    name: test-cluster 11
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
        hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 12
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    aws:
      region: us-west-2 13
      propagateUserTags: true 14
      userTags:
        adminContact: jdoe
        costCenter: 7536
    subnets: 15
      - subnet-1
      - subnet-2
      - subnet-3
    sshKey: ssh-ed25519 AAAA... 16
    pullSecret: '{"auths": ...}' 17

```

1 11 13 17 Required. The installation program prompts you for this value.

2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the *Cloud Credential Operator* entry in the *Red Hat Operators reference* content.

3 6 14 If you do not provide these parameters and values, the installation program provides the default value.

4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

- 5 7 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

- 8 For compute instances running in an AWS Outpost instance, specify a supported instance type in the AWS Outpost instance.
- 9 For compute instances running in AWS Outpost instance, specify the Availability Zone where the Outpost instance is located.
- 10 For compute instances running in AWS Outpost instance, specify volume type gp2, to avoid using gp3 volume type which is not supported.
- 12 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 15 If you provide your own VPC, specify subnets for each availability zone that your cluster uses.
- 16 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

6.16.9. Generating manifest files

Use the installation program to generate a set of manifest files in the assets directory. Manifest files are required to specify the AWS Outposts subnets to use for worker machines, and to specify settings required by the network provider.

If you plan to reuse the **install-config.yaml** file, create a backup file before you generate the manifest files.

Procedure

- Optional: Create a backup copy of the **install-config.yaml** file:

```
$ cp install-config.yaml install-config.yaml.backup
```

- Generate a set of manifests in your assets directory:

```
$ openshift-install create manifests --dir <installation_<_<_directory>
```

This command displays the following messages.

Example output

```
INFO Consuming Install Config from target directory
INFO Manifests created in: <installation_directory>/manifests and
<installation_directory>/openshift
```

The command generates the following manifest files:

Example output

```
$ tree
.
├── manifests
│   ├── cluster-config.yaml
│   ├── cluster-dns-02-config.yml
│   ├── cluster-infrastructure-02-config.yml
│   ├── cluster-ingress-02-config.yml
│   ├── cluster-network-01-crd.yml
│   ├── cluster-network-02-config.yml
│   ├── cluster-proxy-01-config.yaml
│   ├── cluster-scheduler-02-config.yml
│   ├── cvo-overrides.yaml
│   ├── kube-cloud-config.yaml
│   ├── kube-system-configmap-root-ca.yaml
│   ├── machine-config-server-tls-secret.yaml
│   └── openshift-config-secret-pull-secret.yaml
└── openshift
    ├── 99_cloud-creds-secret.yaml
    ├── 99_kubeadmin-password-secret.yaml
    ├── 99_openshift-cluster-api_master-machines-0.yaml
    ├── 99_openshift-cluster-api_master-machines-1.yaml
    ├── 99_openshift-cluster-api_master-machines-2.yaml
    ├── 99_openshift-cluster-api_master-user-data-secret.yaml
    ├── 99_openshift-cluster-api_worker-machineset-0.yaml
    ├── 99_openshift-cluster-api_worker-user-data-secret.yaml
    ├── 99_openshift-machineconfig_99-master-ssh.yaml
    ├── 99_openshift-machineconfig_99-worker-ssh.yaml
    ├── 99_role-cloud-creds-secret-reader.yaml
    └── openshift-install-manifests.yaml
```

6.16.9.1. Modifying manifest files



NOTE

The AWS Outposts environments has the following limitations which require manual modification in the manifest generated files:

- The maximum transmission unit (MTU) of a network connection is the size, in bytes, of the largest permissible packet that can be passed over the connection. The Outpost service link supports a maximum packet size of 1300 bytes. For more information about the service link, see [Outpost connectivity to AWS Regions](#)

You will find more information about how to change these values below.

- Use Outpost Subnet for workers **machineset**
Modify the following file: <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-0.yaml Find the subnet ID and replace it with the ID of the private subnet created in the Outpost. As a result, all the worker machines will be created in the Outpost.
- Specify MTU value for the Network Provider
Outpost service links support a maximum packet size of 1300 bytes. It's required to modify the MTU of the Network Provider to follow this requirement. Create a new file under manifests directory, named cluster-network-03-config.yml

If OpenShift SDN network provider is used, set the MTU value to 1250

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      mtu: 1250
```

If OVN-Kubernetes network provider is used, set the MTU value to 1200

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      mtu: 1200
```

6.16.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.



NOTE

The elevated permissions provided by the **AdministratorAccess** policy are required only during installation.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

6.16.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

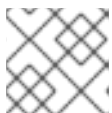
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

6.16.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

6.16.13. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

6.16.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.
- See [About remote health monitoring](#) for more information about the Telemetry service.

6.16.15. Cluster Limitations

IMPORTANT

Network Load Balancer (NLB) and Classic Load Balancer are not supported on AWS Outposts. After the cluster is created, all the Load Balancers are created in the AWS region. In order to use Load Balancers created inside the Outpost instances, Application Load Balancer should be used. The AWS Load Balancer Operator can be used in order to achieve that goal.

If you want to use a public subnet located in the outpost instance for the ALB, you need to remove the special tag (**kubernetes.io/cluster/*-outposts: owned**) that was added earlier during the VPC creation. This will prevent you from creating new Services of type LoadBalancer (Network Load Balancer).

See [Understanding the AWS Load Balancer Operator](#) for more information

IMPORTANT

Persistent storage using AWS Elastic Block Store limitations

- AWS Outposts does not support Amazon Elastic Block Store (EBS) gp3 volumes. After installation, the cluster includes two storage classes - gp3-csi and gp2-csi, with gp3-csi being the default storage class. It is important to always use gp2-csi. You can change the default storage class using the following OpenShift CLI (oc) commands:

```
$ oc annotate --overwrite storageclass gp3-csi storageclass.kubernetes.io/is-
default-class=false
$ oc annotate --overwrite storageclass gp2-csi storageclass.kubernetes.io/is-
default-class=true
```

- To create a Volume in the Outpost instance, the CSI driver determines the Outpost ARN based on the topology keys stored on the CSINode objects. To ensure that the CSI driver uses the correct topology values, it is necessary to use the **WaitForConsumer** volume binding mode and avoid setting allowed topologies on any new storage class created.

6.16.16. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

6.17. UNINSTALLING A CLUSTER ON AWS

You can remove a cluster that you deployed to Amazon Web Services (AWS).

6.17.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

**NOTE**

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

6.17.2. Deleting AWS resources with the Cloud Credential Operator utility

To clean up resources after uninstalling an OpenShift Container Platform cluster with the Cloud Credential Operator (CCO) in manual mode with STS, you can use the CCO utility (**ccoctl**) to remove the AWS resources that **ccoctl** created during installation.

Prerequisites

- Extract and prepare the **ccoctl** binary.
- Install an OpenShift Container Platform cluster with the CCO in manual mode with STS.

Procedure

- Delete the AWS resources that **ccoctl** created:

```
$ ccoctl aws delete \
  --name=<name> \ 1
  --region=<aws_region> 2
```

- 1** **<name>** matches the name that was originally used to create and tag the cloud resources.
- 2** **<aws_region>** is the AWS region in which to delete cloud resources.

Example output:

```
2021/04/08 17:50:41 Identity Provider object .well-known/openid-configuration deleted from
the bucket <name>-oidc
2021/04/08 17:50:42 Identity Provider object keys.json deleted from the bucket <name>-oidc
2021/04/08 17:50:43 Identity Provider bucket <name>-oidc deleted
2021/04/08 17:51:05 Policy <name>-openshift-cloud-credential-operator-cloud-credential-o
associated with IAM Role <name>-openshift-cloud-credential-operator-cloud-credential-o
deleted
2021/04/08 17:51:05 IAM Role <name>-openshift-cloud-credential-operator-cloud-credential-
o deleted
2021/04/08 17:51:07 Policy <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials
associated with IAM Role <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials deleted
2021/04/08 17:51:07 IAM Role <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials
deleted
2021/04/08 17:51:08 Policy <name>-openshift-image-registry-installer-cloud-credentials
associated with IAM Role <name>-openshift-image-registry-installer-cloud-credentials
deleted
2021/04/08 17:51:08 IAM Role <name>-openshift-image-registry-installer-cloud-credentials
deleted
2021/04/08 17:51:09 Policy <name>-openshift-ingress-operator-cloud-credentials associated
with IAM Role <name>-openshift-ingress-operator-cloud-credentials deleted
2021/04/08 17:51:10 IAM Role <name>-openshift-ingress-operator-cloud-credentials deleted
2021/04/08 17:51:11 Policy <name>-openshift-machine-api-aws-cloud-credentials associated
with IAM Role <name>-openshift-machine-api-aws-cloud-credentials deleted
2021/04/08 17:51:11 IAM Role <name>-openshift-machine-api-aws-cloud-credentials deleted
2021/04/08 17:51:39 Identity Provider with ARN arn:aws:iam::<aws_account_id>:oidc-
provider/<name>-oidc.s3.<aws_region>.amazonaws.com deleted
```

Verification

- To verify that the resources are deleted, query AWS. For more information, refer to AWS documentation.

6.17.3. Deleting a cluster with a configured AWS Local Zone infrastructure

After you install a cluster on Amazon Web Services (AWS) into an existing Virtual Private Cloud (VPC), and you set subnets for each Local Zone location, you can delete the cluster and any AWS resources associated with it.

The example in the procedure assumes that you created a VPC and its subnets by using a CloudFormation template.

Prerequisites

- You know the name of the CloudFormation stacks, **<local_zone_stack_name>** and **<vpc_stack_name>**, that were used during the creation of the network. You need the name of the stack to delete the cluster.
- You have access rights to the directory that contains the installation files that were created by the installation program.
- Your account includes a policy that provides you with permissions to delete the CloudFormation stack.

Procedure

1. Change to the directory that contains the stored installation program, and delete the cluster by using the **destroy cluster** command:

```
┌ $ ./openshift-install destroy cluster --dir <installation_directory> \ 1  
└ --log-level=debug 2
```

- 1** For **<installation_directory>**, specify the directory that stored any files created by the installation program.

- 2** To view different log details, specify **error**, **info**, or **warn** instead of **debug**.

2. Delete the CloudFormation stack for the Local Zone subnet:

```
┌ $ aws cloudformation delete-stack --stack-name <local_zone_stack_name>
```

3. Delete the stack of resources that represent the VPC:

```
┌ $ aws cloudformation delete-stack --stack-name <vpc_stack_name>
```

Verification

- Check that you removed the stack resources by issuing the following commands in the AWS CLI. The AWS CLI outputs that no template component exists.

```
┌ $ aws cloudformation describe-stacks --stack-name <local_zone_stack_name>
```

```
┌ $ aws cloudformation describe-stacks --stack-name <vpc_stack_name>
```

Additional resources

- See [Working with stacks](#) in the AWS documentation for more information about AWS CloudFormation stacks.
- [Opt into AWS Local Zones](#)
- [AWS Local Zones available locations](#)
- [AWS Local Zones features](#)

CHAPTER 7. INSTALLING ON AZURE

7.1. PREPARING TO INSTALL ON AZURE

7.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

7.1.2. Requirements for installing OpenShift Container Platform on Azure

Before installing OpenShift Container Platform on Microsoft Azure, you must configure an Azure account. See [Configuring an Azure account](#) for details about account configuration, account limits, public DNS zone configuration, required roles, creating service principals, and supported Azure regions.

If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, see [Manually creating IAM for Azure](#) for other options.

7.1.3. Choosing a method to install OpenShift Container Platform on Azure

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

7.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on Azure infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- **Installing a cluster quickly on Azure** You can install OpenShift Container Platform on Azure infrastructure that is provisioned by the OpenShift Container Platform installation program. You can install a cluster quickly by using the default configuration options.
- **Installing a customized cluster on Azure** You can install a customized cluster on Azure infrastructure that the installation program provisions. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).
- **Installing a cluster on Azure with network customizations** You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- **Installing a cluster on Azure into an existing VNet** You can install OpenShift Container Platform on an existing Azure Virtual Network (VNet) on Azure. You can use this installation method if you have constraints set by the guidelines of your company, such as limits when

creating new accounts or infrastructure.

- **Installing a private cluster on Azure** You can install a private cluster into an existing Azure Virtual Network (VNet) on Azure. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.
- **Installing a cluster on Azure into a government region** OpenShift Container Platform can be deployed into Microsoft Azure Government (MAG) regions that are specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads on Azure.

7.1.3.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on Azure infrastructure that you provision, by using the following method:

- **Installing a cluster on Azure using ARM templates** You can install OpenShift Container Platform on Azure by using infrastructure that you provide. You can use the provided Azure Resource Manager (ARM) templates to assist with an installation.

7.1.4. Next steps

- [Configuring an Azure account](#)

7.2. CONFIGURING AN AZURE ACCOUNT

Before you can install OpenShift Container Platform, you must configure a Microsoft Azure account.



IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

7.2.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default [Azure subscription and service limits, quotas, and constraints](#) affect your ability to install OpenShift Container Platform clusters.



IMPORTANT


Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Default Azure limit	Description
vCPU	44	20 per region	<p>A default cluster requires 44 vCPUs, so you must increase the account limit.</p> <p>By default, each cluster creates the following instances:</p> <ul style="list-style-type: none"> • One bootstrap machine, which is removed after installation • Three control plane machines • Three compute machines <p>Because the bootstrap and control plane machines use Standard_D8s_v3 virtual machines, which use 8 vCPUs, and the compute machines use Standard_D4s_v3 virtual machines, which use 4 vCPUs, a default cluster requires 44 vCPUs. The bootstrap node VM, which uses 8 vCPUs, is used only during installation.</p> <p>To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.</p>
OS Disk	7		<p>Each cluster machine must have a minimum of 100 GB of storage and 300 IOPS. While these are the minimum supported values, faster storage is recommended for production clusters and clusters with intensive workloads. For more information about optimizing storage for performance, see the page titled "Optimizing storage" in the "Scalability and performance" section.</p>
VNet	1	1000 per region	<p>Each default cluster requires one Virtual Network (VNet), which contains two subnets.</p>
Network interfaces	7	65,536 per region	<p>Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.</p>

Component	Number of components required by default	Default Azure limit	Description						
Network security groups	2	5000	<p>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</p> <table border="1"> <tr> <td>control plane</td> <td>Allows the control plane machines to be reached on port 6443 from anywhere</td> </tr> <tr> <td>node</td> <td>Allows worker nodes to be reached from the internet on ports 80 and 443</td> </tr> </table>	control plane	Allows the control plane machines to be reached on port 6443 from anywhere	node	Allows worker nodes to be reached from the internet on ports 80 and 443		
control plane	Allows the control plane machines to be reached on port 6443 from anywhere								
node	Allows worker nodes to be reached from the internet on ports 80 and 443								
Network load balancers	3	1000 per region	<p>Each cluster creates the following load balancers:</p> <table border="1"> <tr> <td>default</td> <td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td> </tr> <tr> <td>internal</td> <td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td> </tr> <tr> <td>external</td> <td>Public IP address that load balances requests to port 6443 across control plane machines</td> </tr> </table> <p>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</p>	default	Public IP address that load balances requests to ports 80 and 443 across worker machines	internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines	external	Public IP address that load balances requests to port 6443 across control plane machines
default	Public IP address that load balances requests to ports 80 and 443 across worker machines								
internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines								
external	Public IP address that load balances requests to port 6443 across control plane machines								
Public IP addresses	3		<p>Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</p>						

Component	Number of components required by default	Default Azure limit	Description
Private IP addresses	7		The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.
Spot VM vCPUs (optional)	0 If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node.	20 per region	<p>This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>Using spot VMs for control plane nodes is not recommended.</p> </div> </div>

Additional resources

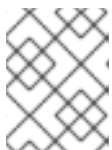
- [Optimizing storage](#).

7.2.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.



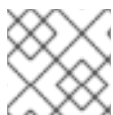
NOTE

For more information about purchasing domains through Azure, see [Buy a custom domain name for Azure App Service](#) in the Azure documentation.

2. If you are using an existing domain and registrar, migrate its DNS to Azure. See [Migrate an active DNS name to Azure App Service](#) in the Azure documentation.
3. Configure DNS for your domain. Follow the steps in the [Tutorial: Host your domain in Azure DNS](#) in the Azure documentation to create a public hosted zone for your domain or subdomain, extract the new authoritative name servers, and update the registrar records for the name servers that your domain uses.
Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
4. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain.

7.2.3. Increasing Azure account limits

To increase an account limit, file a support request on the Azure portal.



NOTE

You can increase only one type of quota per support request.

Procedure

1. From the Azure portal, click **Help + support** in the lower left corner.
2. Click **New support request** and then select the required values:
 - a. From the **Issue type** list, select **Service and subscription limits (quotas)**
 - b. From the **Subscription** list, select the subscription to modify.
 - c. From the **Quota type** list, select the quota to increase. For example, select **Compute-VM (cores-vCPUs) subscription limit increases** to increase the number of vCPUs, which is required to install a cluster.
 - d. Click **Next: Solutions**.
3. On the **Problem Details** page, provide the required information for your quota increase:
 - a. Click **Provide details** and provide the required details in the **Quota details** window.
 - b. In the **SUPPORT METHOD** and **CONTACT INFO** sections, provide the issue severity and your contact details.
4. Click **Next: Review + create** and then click **Create**.

7.2.4. Required Azure roles

OpenShift Container Platform needs a service principal so it can manage Microsoft Azure resources. Before you can create a service principal, review the following information:

Your Azure account subscription must have the following roles:

- **User Access Administrator**
- **Contributor**

Your Azure Active Directory (AD) must have the following permission:

- **"microsoft.directory/servicePrincipals/createAsOwner"**

To set roles on the Azure portal, see the [Manage access to Azure resources using RBAC and the Azure portal](#) in the Azure documentation.

7.2.5. Required Azure permissions for installer-provisioned infrastructure

When you assign **Contributor** and **User Access Administrator** roles to the service principal, you automatically grant all the required permissions.

If your organization's security policies require a more restrictive set of permissions, you can create a [custom role](#) with the necessary permissions. The following permissions are required for creating an OpenShift Container Platform cluster on Microsoft Azure.

Example 7.1. Required permissions for creating authorization resources

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

Example 7.2. Required permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/availabilitySets/write**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**

Example 7.3. Required permissions for creating identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

Example 7.4. Required permissions for creating network resources

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**

- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

NOTE

The following permissions are not required to create the private OpenShift Container Platform cluster on Azure.

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**

Example 7.5. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 7.6. Required permissions for creating a resource group

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

Example 7.7. Required permissions for creating resource tags

- **Microsoft.Resources/tags/write**

Example 7.8. Required permissions for creating storage resources

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

Example 7.9. Optional permissions for creating marketplace virtual machine resources

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

Example 7.10. Optional permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**

Example 7.11. Optional permissions for enabling user-managed encryption

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**

- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

Example 7.12. Optional permissions for installing a private cluster with Azure Network Address Translation (NAT)

- **Microsoft.Network/natGateways/join/action**
- **Microsoft.Network/natGateways/read**
- **Microsoft.Network/natGateways/write**

Example 7.13. Optional permissions for installing a private cluster with Azure firewall

- **Microsoft.Network/azureFirewalls/applicationRuleCollections/write**
- **Microsoft.Network/azureFirewalls/read**
- **Microsoft.Network/azureFirewalls/write**
- **Microsoft.Network/routeTables/join/action**
- **Microsoft.Network/routeTables/read**
- **Microsoft.Network/routeTables/routes/read**
- **Microsoft.Network/routeTables/routes/write**
- **Microsoft.Network/routeTables/write**
- **Microsoft.Network/virtualNetworks/peer/action**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/read**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/write**

Example 7.14. Optional permission for running gather bootstrap

- **Microsoft.Compute/virtualMachines/instanceView/read**

The following permissions are required for deleting an OpenShift Container Platform cluster on Microsoft Azure. You can use the same permissions to delete a private OpenShift Container Platform cluster on Azure.

Example 7.15. Required permissions for deleting authorization resources

- **Microsoft.Authorization/roleAssignments/delete**

Example 7.16. Required permissions for deleting compute resources

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**

Example 7.17. Required permissions for deleting identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

Example 7.18. Required permissions for deleting network resources

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**



NOTE

The following permissions are not required to delete a private OpenShift Container Platform cluster on Azure.

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**

Example 7.19. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 7.20. Required permissions for deleting a resource group

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

Example 7.21. Required permissions for deleting storage resources

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**



NOTE

To install OpenShift Container Platform on Azure, you must scope the permissions to your subscription. Later, you can re-scope these permissions to the installer created resource group. If the public DNS zone is present in a different resource group, then the network DNS zone related permissions must always be applied to your subscription. By default, the OpenShift Container Platform installation program assigns the Azure identity the **Contributor** role.

You can scope all the permissions to your subscription when deleting an OpenShift Container Platform cluster.

7.2.6. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the [Azure CLI](#).
- Your Azure account has the required roles for the subscription that you use.
- If you want to use a custom role, you have created a [custom role](#) with the required permissions listed in the *Required Azure permissions for installer-provisioned infrastructure* section.

Procedure

1. Log in to the Azure CLI:

```
$ az login
```

2. If your Azure account uses subscriptions, ensure that you are using the right subscription:
 - a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": "AzureCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

- b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", 1
  "user": {
    "name": "you@example.com",
```

```
"type": "user"
  }
}
```

- 1 Ensure that the value of the **tenantId** parameter is the correct subscription ID.

c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> 1
```

- 1 Specify the subscription ID.

d. Verify the subscription ID update:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.
4. Create the service principal for your account:

```
$ az ad sp create-for-rbac --role <role_name> \ 1
  --name <service_principal> \ 2
  --scopes /subscriptions/<subscription_id> 3
```

- 1 Defines the role name. You can use the **Contributor** role, or you can specify a custom role which contains the necessary permissions.
- 2 Defines the service principal name.
- 3 Specifies the subscription ID.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
```

include these credentials in your code or check the credentials into your source control. For more information, see <https://aka.ms/azadsp-cli>

```
{
  "appld": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>,
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

- Record the values of the **appld** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.
- If you applied the **Contributor** role to your service principal, assign the **User Administrator Access** role by running the following command:

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appld> --query id -o tsv) 1
```

- Specify the **appld** parameter value for your service principal.

Additional resources

- For more information about CCO modes, see [About the Cloud Credential Operator](#).

7.2.7. Supported Azure Marketplace regions

Installing a cluster using the Azure Marketplace image is available to customers who purchase the offer in North America and EMEA.

While the offer must be purchased in North America or EMEA, you can deploy the cluster to any of the Azure public partitions that OpenShift Container Platform supports.



NOTE

Deploying a cluster using the Azure Marketplace image is not supported for the Azure Government regions.

7.2.8. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription.

Supported Azure public regions

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)

- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **mexicocentral** (Mexico Central)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)

- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

Supported Azure Government regions

Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the [Azure documentation](#). Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

7.2.9. Next steps

- Install an OpenShift Container Platform cluster on Azure. You can [install a customized cluster](#) or [quickly install a cluster](#) with default options.

7.3. MANUALLY CREATING IAM FOR AZURE

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

7.3.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the **credentialsMode** parameter in the **install-config.yaml** file.

If you prefer not to store an administrator-level credential secret in the cluster **kube-system** project, you can set the **credentialsMode** parameter for the CCO to **Manual** when installing OpenShift Container Platform and manage your cloud credentials manually.

Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

Additional resources

- For a detailed description of all available CCO credential modes and their supported platforms, see [About the Cloud Credential Operator](#).

7.3.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. Change to the directory that contains the installation program and create the **install-config.yaml** file by running the following command:

```
$ openshift-install create install-config --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

2. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

3. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

4. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use by running the following command:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

5. Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 \
  --credentials-requests \
  --cloud=azure
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample **CredentialsRequest** object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
    ...
```

6. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample **CredentialsRequest** object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
    ...
  secretRef:
    name: <component-secret>
    namespace: <component-namespace>
  ...
```

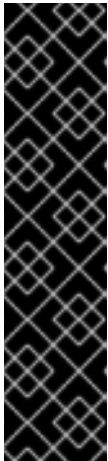
Sample **Secret** object

```
apiVersion: v1
kind: Secret
metadata:
```

```

name: <component-secret>
namespace: <component-namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>

```



IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- To find **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation, run the following command:

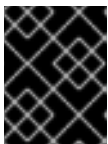
```
$ grep "release.openshift.io/feature-set" *
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```

7. From the directory that contains the installation program, proceed with your cluster creation:

```
$ openshift-install create cluster --dir <installation_directory>
```



IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- [Updating a cluster using the web console](#)
- [Updating a cluster using the CLI](#)

7.3.3. Next steps

- Install an OpenShift Container Platform cluster:

- [Installing a cluster quickly on Azure](#) with default options on installer-provisioned infrastructure
- [Install a cluster with cloud customizations on installer-provisioned infrastructure](#)
- [Install a cluster with network customizations on installer-provisioned infrastructure](#)

7.4. ENABLING USER-MANAGED ENCRYPTION FOR AZURE

In OpenShift Container Platform version 4.12, you can install a cluster with a user-managed encryption key in Azure. To enable this feature, you can prepare an Azure DiskEncryptionSet before installation, modify the `install-config.yaml` file, and then perform post-installation steps.

7.4.1. Preparing an Azure Disk Encryption Set

The OpenShift Container Platform installer can use an existing Disk Encryption Set with a user-managed key. To enable this feature, you can create a Disk Encryption Set in Azure and provide the key to the installer.

Procedure

1. Set the following environment variables for the Azure resource group by running the following command:

```
$ export RESOURCEGROUP="<resource_group>" 1  
LOCATION="<location>" 2
```

- 1** Specifies the name of the Azure resource group where you will create the Disk Encryption Set and encryption key. To avoid losing access to your keys after destroying the cluster, you should create the Disk Encryption Set in a different resource group than the resource group where you install the cluster.

- 2** Specifies the Azure location where you will create the resource group.

2. Set the following environment variables for the Azure Key Vault and Disk Encryption Set by running the following command:

```
$ export KEYVAULT_NAME="<keyvault_name>" 1  
KEYVAULT_KEY_NAME="<keyvault_key_name>" 2  
DISK_ENCRYPTION_SET_NAME="<disk_encryption_set_name>" 3
```

- 1** Specifies the name of the Azure Key Vault you will create.

- 2** Specifies the name of the encryption key you will create.

- 3** Specifies the name of the disk encryption set you will create.

3. Set the environment variable for the ID of your Azure Service Principal by running the following command:

```
$ export CLUSTER_SP_ID="<service_principal_id>" 1
```

1 Specifies the ID of the service principal you will use for this installation.

4. Enable host-level encryption in Azure by running the following commands:

```
$ az feature register --namespace "Microsoft.Compute" --name "EncryptionAtHost"
```

```
$ az feature show --namespace Microsoft.Compute --name EncryptionAtHost
```

```
$ az provider register -n Microsoft.Compute
```

5. Create an Azure Resource Group to hold the disk encryption set and associated resources by running the following command:

```
$ az group create --name $RESOURCEGROUP --location $LOCATION
```

6. Create an Azure key vault by running the following command:

```
$ az keyvault create -n $KEYVAULT_NAME -g $RESOURCEGROUP -l $LOCATION \
--enable-purge-protection true
```

7. Create an encryption key in the key vault by running the following command:

```
$ az keyvault key create --vault-name $KEYVAULT_NAME -n $KEYVAULT_KEY_NAME \
--protection software
```

8. Capture the ID of the key vault by running the following command:

```
$ KEYVAULT_ID=$(az keyvault show --name $KEYVAULT_NAME --query "[id]" -o tsv)
```

9. Capture the key URL in the key vault by running the following command:

```
$ KEYVAULT_KEY_URL=$(az keyvault key show --vault-name $KEYVAULT_NAME --name \
$KEYVAULT_KEY_NAME --query "[key.kid]" -o tsv)
```

10. Create a disk encryption set by running the following command:

```
$ az disk-encryption-set create -n $DISK_ENCRYPTION_SET_NAME -l $LOCATION -g \
$RESOURCEGROUP --source-vault $KEYVAULT_ID --key-url $KEYVAULT_KEY_URL
```

11. Grant the DiskEncryptionSet resource access to the key vault by running the following commands:

```
$ DES_IDENTITY=$(az disk-encryption-set show -n $DISK_ENCRYPTION_SET_NAME -g \
$RESOURCEGROUP --query "[identity.principalId]" -o tsv)
```

```
$ az keyvault set-policy -n $KEYVAULT_NAME -g $RESOURCEGROUP --object-id \
$DES_IDENTITY --key-permissions wrapkey unwrapkey get
```

12. Grant the Azure Service Principal permission to read the DiskEncryptionSet by running the following commands:

```
$ DES_RESOURCE_ID=$(az disk-encryption-set show -n
$DISK_ENCRYPTION_SET_NAME -g \
  $RESOURCEGROUP --query "[id]" -o tsv)
```

```
$ az role assignment create --assignee $CLUSTER_SP_ID --role "<reader_role>" \
  --scope $DES_RESOURCE_ID -o jsonc
```

- 1 Specifies an Azure role with read permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.

7.4.2. Next steps

- Install an OpenShift Container Platform cluster:
 - [Install a cluster with customizations on installer-provisioned infrastructure](#)
 - [Install a cluster with network customizations on installer-provisioned infrastructure](#)
 - [Install a cluster into an existing VNet on installer-provisioned infrastructure](#)
 - [Install a private cluster on installer-provisioned infrastructure](#)
 - [Install a cluster into an government region on installer-provisioned infrastructure](#)

7.5. INSTALLING A CLUSTER QUICKLY ON AZURE

In OpenShift Container Platform version 4.12, you can install a cluster on Microsoft Azure that uses the default configuration options.

7.5.1. Prerequisites

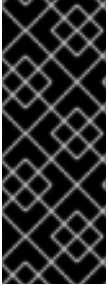
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

7.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

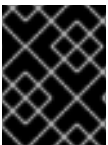
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.5.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

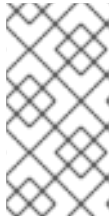
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.5.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

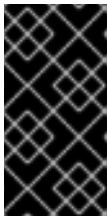
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.5.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Provide values at the prompts:
 - a. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- b. Select **azure** as the platform to target.
- c. If the installation program cannot locate the **osServicePrincipal.json** configuration file,

which contains Microsoft Azure profile information, in the `~/.azure/` directory on your computer, the installer prompts you to specify the following Azure parameter values for your subscription and service principal.

- **azure subscription id** The subscription ID to use for the cluster. Specify the **id** value in your account output.
- **azure tenant id** The tenant ID. Specify the **tenantid** value in your account output.
- **azure service principal client id** The value of the **appid** parameter for the service principal.
- **azure service principal client secret** The value of the **password** parameter for the service principal.



IMPORTANT

After you enter values for the previously listed parameters, the installation program creates a **osServicePrincipal.json** configuration file and stores this file in the `~/.azure/` directory on your computer. These actions ensure that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

- Select the region to deploy the cluster to.
- Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.
- Enter a descriptive name for your cluster.



IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

- Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.5.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

7.5.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.5.8. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.5.9. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

7.6. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on infrastructure that the installation program provisions on Microsoft Azure. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

7.6.1. Prerequisites

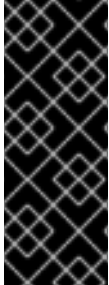
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .
- If you use customer-managed encryption keys, you [prepared your Azure environment for encryption](#).

7.6.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.6.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

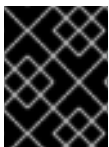
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.6.4. Selecting an Azure Marketplace image

If you are deploying an OpenShift Container Platform cluster using the Azure Marketplace offering, you must first obtain the Azure Marketplace image. The installation program uses this image to deploy worker nodes. When obtaining your image, consider the following:

- While the images are the same, the Azure Marketplace publisher is different depending on your region. If you are located in North America, specify **redhat** as the publisher. If you are located in EMEA, specify **redhat-limited** as the publisher.
- The offer includes a **rh-ocp-worker** SKU and a **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker** SKU represents a Hyper-V generation version 2 VM image. The default instance types used in OpenShift Container Platform are version 2 compatible. If you plan to use an instance type that is only version 1 compatible, use the image associated with the **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker-gen1** SKU represents a Hyper-V version 1 VM image.



IMPORTANT

Installing images with the Azure marketplace is not supported on clusters with 64-bit ARM instances.

Prerequisites

- You have installed the Azure CLI client (**az**).
- Your Azure account is entitled for the offer and you have logged into this account with the Azure CLI client.

Procedure

1. Display all of the available OpenShift Container Platform images by running one of the following commands:

- North America:

```
$ az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
```

Example output

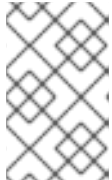
Offer	Publisher	Sku	Urn	Version
rh-ocp-worker-ocpworker:4.8.2021122100	RedHat	rh-ocp-worker	RedHat:rh-ocp-worker:rh-ocp-worker:4.8.2021122100	
rh-ocp-worker-gen1:4.8.2021122100	RedHat	rh-ocp-worker-gen1	RedHat:rh-ocp-worker:rh-ocp-worker-gen1:4.8.2021122100	

- EMEA:

```
$ az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

Example output

Offer	Publisher	Sku	Urn	Version
rh-ocp-worker	redhat-limited	rh-ocp-worker	redhat-limited:rh-ocp-worker:rh-ocp-worker:4.8.2021122100	4.8.2021122100
rh-ocp-worker-gen1	redhat-limited	rh-ocp-worker-gen1	redhat-limited:rh-ocp-worker:rh-ocp-worker-gen1:4.8.2021122100	4.8.2021122100

**NOTE**

Regardless of the version of OpenShift Container Platform that you install, the correct version of the Azure Marketplace image to use is 4.8. If required, your VMs are automatically upgraded as part of the installation process.

2. Inspect the image for your offer by running one of the following commands:

- North America:

```
$ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. Review the terms of the offer by running one of the following commands:

- North America:

```
$ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

4. Accept the terms of the offering by running one of the following commands:

- North America:

```
$ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

5. Record the image details of your offer. You must update the **compute** section in the **install-config.yaml** file with values for **publisher**, **offer**, **sku**, and **version** before deploying the cluster.

Sample install-config.yaml file with the Azure Marketplace worker nodes

```
apiVersion: v1
```

```

baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
  azure:
    type: Standard_D4s_v5
    osImage:
      publisher: redhat
      offer: rh-ocp-worker
      sku: rh-ocp-worker
      version: 4.8.2021122100
  replicas: 3

```

7.6.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

- Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
- Select your infrastructure provider.
- Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

- Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull

secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.6.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **azure** as the platform to target.
- iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

- **azure subscription id** The subscription ID to use for the cluster. Specify the **id** value in your account output.
 - **azure tenant id** The tenant ID. Specify the **tenantId** value in your account output.
 - **azure service principal client id** The value of the **appId** parameter for the service principal.
 - **azure service principal client secret** The value of the **password** parameter for the service principal.
- iv. Select the region to deploy the cluster to.
 - v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.
 - vi. Enter a descriptive name for your cluster.



IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

- vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

7.6.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.6.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.1. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

7.6.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 7.2. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


7.6.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 7.3. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

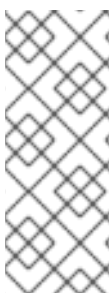
Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 898 593 1279" data-label="Image"> </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 1323 593 1671" data-label="Image"> </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="671 107 863 141">IMPORTANT</p> <p data-bbox="671 181 932 909">To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="671 965 762 999">NOTE</p> <p data-bbox="671 1032 922 1151">If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p>  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

7.6.6.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.



NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within [a region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.4. Additional Azure parameters

Parameter	Description	Values
compute.platform.azure.encryptionAtHost	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
compute.platform.azure.osDisk.diskSize GB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS , premium_LRS , or standardSSD_LRS . The default is premium_LRS .
compute.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	Enabled , Disabled . The default is Disabled .
compute.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
compute.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
compute.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .

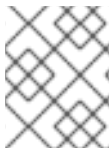
Parameter	Description	Values
<code>compute.platform.azure.vmNetworkingType</code>	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
<code>compute.platform.azure.type</code>	Defines the Azure instance type for compute machines.	String
<code>compute.platform.azure.zones</code>	The availability zones where the installation program creates compute machines.	String list
<code>controlPlane.platform.azure.type</code>	Defines the Azure instance type for control plane machines.	String
<code>controlPlane.platform.azure.zones</code>	The availability zones where the installation program creates control plane machines.	String list
<code>platform.azure.defaultMachinePlatform.enableEncryptionAtHost</code>	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
<code>platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.name</code>	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, production_disk_encryption_set .

Parameter	Description	Values
<code>platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.resourceGroup</code>	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, production_encryption_resource_group .
<code>platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.subscriptionId</code>	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
<code>platform.azure.defaultMachinePlatform.ossDisk.diskSizeGB</code>	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
<code>platform.azure.defaultMachinePlatform.ossDisk.diskType</code>	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
<code>platform.azure.defaultMachinePlatform.type</code>	The Azure instance type for control plane and compute machines.	The Azure instance type.
<code>platform.azure.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates compute and control plane machines.	String list.
<code>controlPlane.platform.azure.encryptionAtHost</code>	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .

Parameter	Description	Values
controlPlane.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
controlPlane.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
controlPlane.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
controlPlane.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
controlPlane.platform.azure.vmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.region	The name of the Azure region that hosts your cluster.	Any valid region name, such as centralus .
platform.azure.zone	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example ["1", "2", "3"] .
platform.azure.defaultMachinePlatform.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
platform.azure.networkResourceGroupName	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName .	String.
platform.azure.virtualNetwork	The name of the existing VNet that you want to deploy your cluster to.	String.

Parameter	Description	Values
platform.azure.controlPlaneSubnet	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.computeSubnet	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.	Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud .
platform.azure.defaultMachinePlatform.vmmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	Accelerated or Basic . If instance type of control plane and compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .

**NOTE**

You cannot customize [Azure Availability Zones](#) or [Use tags to organize your Azure resources](#) with an Azure cluster.

7.6.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 7.5. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.6.6.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 7.22. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- r6i.*
- t3.*
- t3a.*

7.6.6.4. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 7.23. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

7.6.6.5. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
    osDisk:
      diskSizeGB: 1024 5
      diskType: Premium_LRS
      diskEncryptionSet:
        resourceGroup: disk_encryption_set_resource_group
        name: disk_encryption_set_name
        subscriptionId: secondary_subscription_id
      type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        ultraSSDCapability: Enabled

```

```

type: Standard_D2s_v3
encryptionAtHost: true
osDisk:
  diskSizeGB: 512 8
  diskType: Standard_LRS
  diskEncryptionSet:
    resourceGroup: disk_encryption_set_resource_group
    name: disk_encryption_set_name
    subscriptionId: secondary_subscription_id
zones: 9
- "1"
- "2"
- "3"
replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 12
    region: centralus 13
    resourceGroupName: existing_resource_group 14
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 15
fips: false 16
sshKey: ssh-ed25519 AAAA... 17

```

1 10 13 15 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

- 5 8 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 9 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 Specify the name of the resource group that contains the DNS zone for your base domain.
- 14 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 16 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 17 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

7.6.6.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of

them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

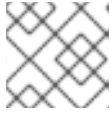
For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

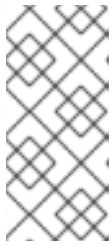
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see [Accelerated Networking for Microsoft Azure VMs](#).

7.6.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

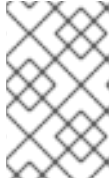
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



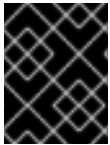
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.

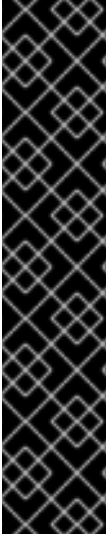


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.6.8. Finalizing user-managed encryption after installation

If you installed OpenShift Container Platform using a user-managed encryption key, you can complete the installation by creating a new storage class and granting write permissions to the Azure cluster resource group.

Procedure

1. Obtain the identity of the cluster resource group used by the installer:
 - a. If you specified an existing resource group in **install-config.yaml**, obtain its Azure identity by running the following command:

```
$ az identity list --resource-group "<existing_resource_group>"
```

- b. If you did not specify a existing resource group in **install-config.yaml**, locate the resource group that the installer created, and then obtain its Azure identity by running the following commands:

```
$ az group list
```

```
$ az identity list --resource-group "<installer_created_resource_group>"
```

2. Grant a role assignment to the cluster resource group so that it can write to the Disk Encryption Set by running the following command:

```
$ az role assignment create --role "<privileged_role>" 1  
--assignee "<resource_group_identity>" 2
```

- 1** Specifies an Azure role that has read/write permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.
- 2** Specifies the identity of the cluster resource group.

3. Obtain the **id** of the disk encryption set you created prior to installation by running the following command:

```
$ az disk-encryption-set show -n <disk_encryption_set_name> \ 1
--resource-group <resource_group_name> 2
```

- 1 Specifies the name of the disk encryption set.
- 2 Specifies the resource group that contains the disk encryption set. The **id** is in the format of **"/subscriptions/.../resourceGroups/.../providers/Microsoft.Compute/diskEncryptionSets/..."**.

4. Obtain the identity of the cluster service principal by running the following command:

```
$ az identity show -g <cluster_resource_group> \ 1
-n <cluster_service_principal_name> \ 2
--query principalId --out tsv
```

- 1 Specifies the name of the cluster resource group created by the installation program.
- 2 Specifies the name of the cluster service principal created by the installation program. The identity is in the format of **12345678-1234-1234-1234-1234567890**.

5. Create a role assignment that grants the cluster service principal necessary privileges to the disk encryption set by running the following command:

```
$ az role assignment create --assignee <cluster_service_principal_id> \ 1
--role <privileged_role> \ 2
--scope <disk_encryption_set_id> \ 3
```

- 1 Specifies the ID of the cluster service principal obtained in the previous step.
- 2 Specifies the Azure role name. You can use the **Contributor** role or a custom role with the necessary permissions.
- 3 Specifies the ID of the disk encryption set.

6. Create a storage class that uses the user-managed disk encryption set:

- a. Save the following storage class definition to a file, for example **storage-class-definition.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: managed-premium
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
  kind: Managed
  diskEncryptionSetID: "<disk_encryption_set_ID>" 1
  resourceGroup: "<resource_group_name>" 2
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: WaitForFirstConsumer
```

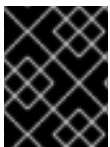
- - 1 Specifies the ID of the disk encryption set that you created in the prerequisite steps, for example `"/subscriptions/xxxxxx-xxxxx-xxxxx/resourceGroups/test-encryption/providers/Microsoft.Compute/diskEncryptionSets/disk-encryption-set-xxxxxx"`.
 - 2 Specifies the name of the resource group used by the installer. This is the same resource group from the first step.
- b. Create the storage class **managed-premium** from the file you created by running the following command:

```
$ oc create -f storage-class-definition.yaml
```

7. Select the **managed-premium** storage class when you create persistent volumes to use encrypted storage.

7.6.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

7.6.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.6.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.6.12. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

7.7. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Microsoft Azure. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

7.7.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#). Manual mode can also be used in environments where the cloud IAM APIs are not reachable.
- If you use customer-managed encryption keys, you [prepared your Azure environment for encryption](#).

7.7.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

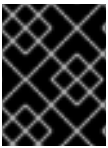
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.7.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.7.4. Obtaining the installation program

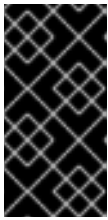
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

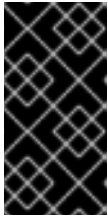
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.7.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following

- a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
- Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- Select **azure** as the platform to target.
- If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:
 - **azure subscription id** The subscription ID to use for the cluster. Specify the **id** value in your account output.
 - **azure tenant id** The tenant ID. Specify the **tenantId** value in your account output.
 - **azure service principal client id** The value of the **appId** parameter for the service principal.
 - **azure service principal client secret** The value of the **password** parameter for the service principal.
- Select the region to deploy the cluster to.
- Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.
- Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

- vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

7.7.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.7.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.6. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

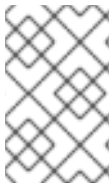
Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

7.7.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 7.7. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


7.7.5.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 7.8. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="488 192 593 913"></p> <p data-bbox="671 107 935 913">IMPORTANT To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="488 958 593 1155"></p> <p data-bbox="671 965 935 1155">NOTE If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	The SSH key to authenticate access to your cluster machines.  NOTE For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.	For example, sshKey: ssh-ed25519 AAAA...

7.7.5.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.



NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within a [region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.9. Additional Azure parameters

Parameter	Description	Values
compute.platform.azure.encryptionAtHost	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
compute.platform.azure.osDisk.diskSize GB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS , premium_LRS , or standardSSD_LRS . The default is premium_LRS .
compute.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	Enabled , Disabled . The default is Disabled .
compute.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
compute.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
compute.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .

Parameter	Description	Values
<code>compute.platform.azure.vmNetworkingType</code>	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
<code>compute.platform.azure.type</code>	Defines the Azure instance type for compute machines.	String
<code>compute.platform.azure.zones</code>	The availability zones where the installation program creates compute machines.	String list
<code>controlPlane.platform.azure.type</code>	Defines the Azure instance type for control plane machines.	String
<code>controlPlane.platform.azure.zones</code>	The availability zones where the installation program creates control plane machines.	String list
<code>platform.azure.defaultMachinePlatform.enableEncryptionAtHost</code>	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
<code>platform.azure.defaultMachinePlatform.ostDisk.diskEncryptionSet.name</code>	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, production_disk_encryption_set .

Parameter	Description	Values
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, production_encryption_resource_group .
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
platform.azure.defaultMachinePlatform.ossDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.ossDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.defaultMachinePlatform.zones	The availability zones where the installation program creates compute and control plane machines.	String list.
controlPlane.platform.azure.encryptionAtHost	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .

Parameter	Description	Values
controlPlane.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
controlPlane.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
controlPlane.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
controlPlane.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
controlPlane.platform.azure.vmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.region	The name of the Azure region that hosts your cluster.	Any valid region name, such as centralus .
platform.azure.zone	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example ["1", "2", "3"] .
platform.azure.defaultMachinePlatform.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
platform.azure.networkResourceGroupName	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName .	String.
platform.azure.virtualNetwork	The name of the existing VNet that you want to deploy your cluster to.	String.

Parameter	Description	Values
platform.azure.controlPlaneSubnet	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.computeSubnet	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.	Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud .
platform.azure.defaultMachinePlatform.vmmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	Accelerated or Basic . If instance type of control plane and compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .

**NOTE**

You cannot customize [Azure Availability Zones](#) or [Use tags to organize your Azure resources](#) with an Azure cluster.

7.7.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 7.10. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.7.5.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 7.24. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- **r6i.***
- **t3.***
- **t3a.***

7.7.5.4. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 7.25. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

7.7.5.5. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
    osDisk:
      diskSizeGB: 1024 5
      diskType: Premium_LRS
      diskEncryptionSet:
        resourceGroup: disk_encryption_set_resource_group
        name: disk_encryption_set_name
        subscriptionId: secondary_subscription_id
      type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        ultraSSDCapability: Enabled

```



```

type: Standard_D2s_v3
encryptionAtHost: true
osDisk:
  diskSizeGB: 512 8
  diskType: Standard_LRS
  diskEncryptionSet:
    resourceGroup: disk_encryption_set_resource_group
    name: disk_encryption_set_name
    subscriptionId: secondary_subscription_id
zones: 9
- "1"
- "2"
- "3"
replicas: 5
metadata:
  name: test-cluster 10
networking: 11
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: centralus 14
    resourceGroupName: existing_resource_group 15
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 16
fips: false 17
sshKey: ssh-ed25519 AAAA... 18

```

1 10 14 16 Required. The installation program prompts you for this value.

2 6 11 If you do not provide these parameters and values, the installation program provides the default value.

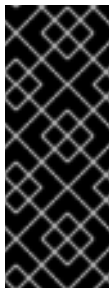
3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

- 5 8 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 9 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.
- 12 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 13 Specify the name of the resource group that contains the DNS zone for your base domain.
- 15 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 17 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 18 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

7.7.5.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of

them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

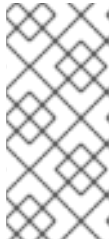
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

7.7.6. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

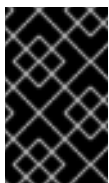
You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.

**NOTE**

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

**IMPORTANT**

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

7.7.7. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

- Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

- Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

```

defaultNetwork:
  openshiftSDNConfig:
    vxlanPort: 4800

```

Enable IPsec for the OVN-Kubernetes network provider

```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}

```

- Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

7.7.8. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

7.7.8.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 7.11. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .


Field	Type	Description
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 7.12. **defaultNetwork** object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 7.13. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>

Field	Type	Description
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 7.14. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.

Field	Type	Description
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <p> NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 7.15. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 7.16. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 7.17. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

7.7.9. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.



IMPORTANT

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yaml** in the **<installation_directory>/manifests/** directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yaml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  EOF
```

where:

<installation_directory>

Specifies the directory name that contains the **manifests/** directory for your cluster.

3. Open the **cluster-network-03-config.yaml** file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

Specify a hybrid networking configuration

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
        hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

- 1 Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR cannot overlap with the **clusterNetwork** CIDR.
- 2 Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **4789** port. For more information on this requirement, see the Microsoft documentation on [Pod-to-pod connectivity between hosts is broken](#).

**NOTE**

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.
5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.



NOTE

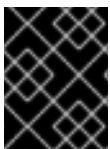
For more information on using Linux and Windows nodes in the same cluster, see [Understanding Windows container workloads](#).

Additional resources

- For more details about Accelerated Networking, see [Accelerated Networking for Microsoft Azure VMs](#).

7.7.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

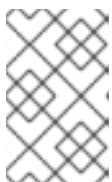
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.7.11. Finalizing user-managed encryption after installation

If you installed OpenShift Container Platform using a user-managed encryption key, you can complete the installation by creating a new storage class and granting write permissions to the Azure cluster resource group.

Procedure

1. Obtain the identity of the cluster resource group used by the installer:
 - a. If you specified an existing resource group in **install-config.yaml**, obtain its Azure identity by running the following command:

```
$ az identity list --resource-group "<existing_resource_group>"
```

- b. If you did not specify an existing resource group in **install-config.yaml**, locate the resource group that the installer created, and then obtain its Azure identity by running the following commands:

```
$ az group list
```

```
$ az identity list --resource-group "<installer_created_resource_group>"
```

2. Grant a role assignment to the cluster resource group so that it can write to the Disk Encryption Set by running the following command:

```
$ az role assignment create --role "<privileged_role>" \
  --assignee "<resource_group_identity>"
```

- 1 Specifies an Azure role that has read/write permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.

- 2 Specifies the identity of the cluster resource group.

3. Obtain the **id** of the disk encryption set you created prior to installation by running the following command:

```
$ az disk-encryption-set show -n <disk_encryption_set_name> \
  --resource-group <resource_group_name>
```

- 1 Specifies the name of the disk encryption set.

- 2 Specifies the resource group that contains the disk encryption set. The **id** is in the format of **"/subscriptions/.../resourceGroups/.../providers/Microsoft.Compute/diskEncryptionSets/..."**.

4. Obtain the identity of the cluster service principal by running the following command:

```
$ az identity show -g <cluster_resource_group> \
  -n <cluster_service_principal_name> \
  --query principalId --out tsv
```

- 1 Specifies the name of the cluster resource group created by the installation program.

- 2 Specifies the name of the cluster service principal created by the installation program. The identity is in the format of **12345678-1234-1234-1234-1234567890**.

5. Create a role assignment that grants the cluster service principal necessary privileges to the disk encryption set by running the following command:

```
$ az role assignment create --assignee <cluster_service_principal_id> \
  --role <privileged_role> \
  --scope <disk_encryption_set_id>
```

- 1 Specifies the ID of the cluster service principal obtained in the previous step.
- 2 Specifies the Azure role name. You can use the **Contributor** role or a custom role with the necessary permissions.
- 3 Specifies the ID of the disk encryption set.

6. Create a storage class that uses the user-managed disk encryption set:

- a. Save the following storage class definition to a file, for example **storage-class-definition.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: managed-premium
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
  kind: Managed
  diskEncryptionSetID: "<disk_encryption_set_ID>" 1
  resourceGroup: "<resource_group_name>" 2
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: WaitForFirstConsumer
```

- 1 Specifies the ID of the disk encryption set that you created in the prerequisite steps, for example **"/subscriptions/xxxxxx-xxxxx-xxxxx/resourceGroups/test-encryption/providers/Microsoft.Compute/diskEncryptionSets/disk-encryption-set-xxxxxx"**.
- 2 Specifies the name of the resource group used by the installer. This is the same resource group from the first step.

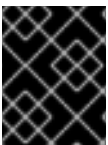
- b. Create the storage class **managed-premium** from the file you created by running the following command:

```
$ oc create -f storage-class-definition.yaml
```

7. Select the **managed-premium** storage class when you create persistent volumes to use encrypted storage.

7.7.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

7.7.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.7.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.7.15. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

7.8. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET

In OpenShift Container Platform version 4.12, you can install a cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

7.8.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).
- If you use customer-managed encryption keys, you [prepared your Azure environment for encryption](#).

7.8.2. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.12, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

7.8.2.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICs for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.



NOTE

By default, if you specify availability zones in the `install-config.yaml` file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within [a region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.



NOTE

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

7.8.2.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.



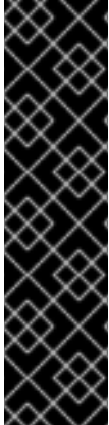
IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 7.18. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	

Port	Description	Control plane	Compute
22623	Allows internal communication to the machine config server for provisioning machines	x	



IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Additional resources

- [About the OpenShift SDN network plugin](#)

7.8.2.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

7.8.2.3. Isolation between clusters

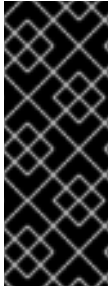
Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

7.8.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

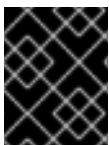
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.8.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

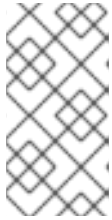
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.8.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

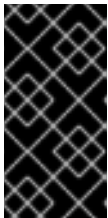
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.8.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **azure** as the platform to target.
- iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:
 - **azure subscription id** The subscription ID to use for the cluster. Specify the **id** value in your account output.
 - **azure tenant id** The tenant ID. Specify the **tenantId** value in your account output.
 - **azure service principal client id** The value of the **appId** parameter for the service principal.
 - **azure service principal client secret** The value of the **password** parameter for the service principal.

- iv. Select the region to deploy the cluster to.
- v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.
- vi. Enter a descriptive name for your cluster.



IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

- vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

7.8.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.8.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.19. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform.<platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

7.8.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 7.20. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


7.8.6.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:



Table 7.21. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

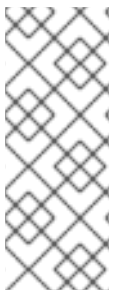
Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="671 107 863 141">IMPORTANT</p>  <p data-bbox="671 181 932 909">To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="671 965 762 999">NOTE</p>  <p data-bbox="671 1032 922 1155">If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p>  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

7.8.6.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.



NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within a [region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.22. Additional Azure parameters

Parameter	Description	Values
compute.platform.azure.encryptionAtHost	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
compute.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS , premium_LRS , or standardSSD_LRS . The default is premium_LRS .
compute.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	Enabled , Disabled . The default is Disabled .
compute.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
compute.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
compute.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .

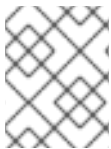
Parameter	Description	Values
<code>compute.platform.azure.vmNetworkingType</code>	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
<code>compute.platform.azure.type</code>	Defines the Azure instance type for compute machines.	String
<code>compute.platform.azure.zones</code>	The availability zones where the installation program creates compute machines.	String list
<code>controlPlane.platform.azure.type</code>	Defines the Azure instance type for control plane machines.	String
<code>controlPlane.platform.azure.zones</code>	The availability zones where the installation program creates control plane machines.	String list
<code>platform.azure.defaultMachinePlatform.enableEncryptionAtHost</code>	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
<code>platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.name</code>	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, production_disk_encryption_set .

Parameter	Description	Values
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, production_encryption_resource_group .
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
platform.azure.defaultMachinePlatform.ossDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.ossDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.defaultMachinePlatform.zones	The availability zones where the installation program creates compute and control plane machines.	String list.
controlPlane.platform.azure.encryptionAtHost	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .

Parameter	Description	Values
controlPlane.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
controlPlane.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
controlPlane.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
controlPlane.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
controlPlane.platform.azure.vmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.region	The name of the Azure region that hosts your cluster.	Any valid region name, such as centralus .
platform.azure.zone	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example ["1", "2", "3"] .
platform.azure.defaultMachinePlatform.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
platform.azure.networkResourceGroupName	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName .	String.
platform.azure.virtualNetwork	The name of the existing VNet that you want to deploy your cluster to.	String.

Parameter	Description	Values
platform.azure.controlPlaneSubnet	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.computeSubnet	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.	Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud .
platform.azure.defaultMachinePlatform.vmmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	Accelerated or Basic . If instance type of control plane and compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .

**NOTE**

You cannot customize [Azure Availability Zones](#) or [Use tags to organize your Azure resources](#) with an Azure cluster.

7.8.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 7.23. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.8.6.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 7.26. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- r6i.*
- t3.*
- t3a.*

7.8.6.4. Tested instance types for Azure on 64-bit ARM infrastructures

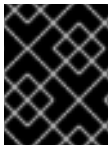
The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 7.27. Machine types based on 64-bit ARM architecture

- c6g.*
- m6g.*

7.8.6.5. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
    osDisk:
      diskSizeGB: 1024 5
      diskType: Premium_LRS
      diskEncryptionSet:
        resourceGroup: disk_encryption_set_resource_group
        name: disk_encryption_set_name
        subscriptionId: secondary_subscription_id
      type: Standard_D8s_v3
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      azure:
        ultraSSDCapability: Enabled

```

```

type: Standard_D2s_v3
encryptionAtHost: true
osDisk:
  diskSizeGB: 512 8
  diskType: Standard_LRS
  diskEncryptionSet:
    resourceGroup: disk_encryption_set_resource_group
    name: disk_encryption_set_name
    subscriptionId: secondary_subscription_id
  zones: 9
  - "1"
  - "2"
  - "3"
replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 12
    region: centralus 13
    resourceGroupName: existing_resource_group 14
    networkResourceGroupName: vnet_resource_group 15
    virtualNetwork: vnet 16
    controlPlaneSubnet: control_plane_subnet 17
    computeSubnet: compute_subnet 18
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
  pullSecret: '{"auths": ...}' 19
  fips: false 20
  sshKey: ssh-ed25519 AAAA... 21

```

1 10 13 19 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous

can disable it by setting the parameter value to `disabled`. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

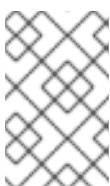
- 5 8 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 9 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 Specify the name of the resource group that contains the DNS zone for your base domain.
- 14 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 15 If you use an existing VNet, specify the name of the resource group that contains it.
- 16 If you use an existing VNet, specify its name.
- 17 If you use an existing VNet, specify the name of the subnet to host the control plane machines.
- 18 If you use an existing VNet, specify the name of the subnet to host the compute machines.
- 20 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 21 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

7.8.6.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

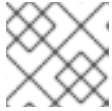
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the

trustedCA field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

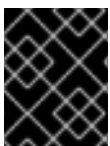
Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see [Accelerated Networking for Microsoft Azure VMs](#).

7.8.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.8.8. Finalizing user-managed encryption after installation

If you installed OpenShift Container Platform using a user-managed encryption key, you can complete the installation by creating a new storage class and granting write permissions to the Azure cluster resource group.

Procedure

1. Obtain the identity of the cluster resource group used by the installer:
 - a. If you specified an existing resource group in **install-config.yaml**, obtain its Azure identity by running the following command:

```
$ az identity list --resource-group "<existing_resource_group>"
```

- b. If you did not specify a existing resource group in **install-config.yaml**, locate the resource group that the installer created, and then obtain its Azure identity by running the following commands:

```
$ az group list
```

```
$ az identity list --resource-group "<installer_created_resource_group>"
```

2. Grant a role assignment to the cluster resource group so that it can write to the Disk Encryption Set by running the following command:

```
$ az role assignment create --role "<privileged_role>" \ 1  
--assignee "<resource_group_identity>" 2
```

- 1** Specifies an Azure role that has read/write permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.
- 2** Specifies the identity of the cluster resource group.

3. Obtain the **id** of the disk encryption set you created prior to installation by running the following command:

```
$ az disk-encryption-set show -n <disk_encryption_set_name> \ 1
--resource-group <resource_group_name> 2
```

- 1 Specifies the name of the disk encryption set.
- 2 Specifies the resource group that contains the disk encryption set. The **id** is in the format of **"/subscriptions/.../resourceGroups/.../providers/Microsoft.Compute/diskEncryptionSets/..."**.

4. Obtain the identity of the cluster service principal by running the following command:

```
$ az identity show -g <cluster_resource_group> \ 1
-n <cluster_service_principal_name> \ 2
--query principalId --out tsv
```

- 1 Specifies the name of the cluster resource group created by the installation program.
- 2 Specifies the name of the cluster service principal created by the installation program. The identity is in the format of **12345678-1234-1234-1234-1234567890**.

5. Create a role assignment that grants the cluster service principal necessary privileges to the disk encryption set by running the following command:

```
$ az role assignment create --assignee <cluster_service_principal_id> \ 1
--role <privileged_role> \ 2
--scope <disk_encryption_set_id> \ 3
```

- 1 Specifies the ID of the cluster service principal obtained in the previous step.
- 2 Specifies the Azure role name. You can use the **Contributor** role or a custom role with the necessary permissions.
- 3 Specifies the ID of the disk encryption set.

6. Create a storage class that uses the user-managed disk encryption set:

- a. Save the following storage class definition to a file, for example **storage-class-definition.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: managed-premium
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
  kind: Managed
  diskEncryptionSetID: "<disk_encryption_set_ID>" 1
  resourceGroup: "<resource_group_name>" 2
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: WaitForFirstConsumer
```



```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

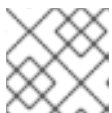
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

7.8.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.8.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.8.12. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

7.9. INSTALLING A PRIVATE CLUSTER ON AZURE

In OpenShift Container Platform version 4.12, you can install a private cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

7.9.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .
- If you use customer-managed encryption keys, you [prepared your Azure environment for encryption](#).

7.9.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.
 - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

7.9.2.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder to resolve the cluster's private DNS records. The cluster's machines use **168.63.129.16** internally for DNS resolution. For more information, see [What is Azure Private DNS?](#) and [What is IP address 168.63.129.16?](#) in the Azure documentation.

The cluster still requires access to internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A **BaseDomainResourceGroup**, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

7.9.2.1.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

7.9.2.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the **install-config.yaml** file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.

You must ensure the following items are available before setting user-defined routing:

- Egress to the internet is possible to pull container images, unless using an OpenShift image registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the *Configuring your firewall* section.

There are several pre-existing networking setups that are supported for internet access using user-defined routing.

Private cluster with network address translation

You can use [Azure VNET network address translation \(NAT\)](#) to provide outbound internet access for the subnets in your cluster. You can reference [Create a NAT gateway using Azure CLI](#) in the Azure documentation for configuration instructions.

When using a VNet setup with Azure NAT and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with Azure Firewall

You can use Azure Firewall to provide outbound routing for the VNet used to install the cluster. You can learn more about [providing user-defined routing with Azure Firewall](#) in the Azure documentation.

When using a VNet setup with Azure Firewall and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with a proxy configuration

You can use a proxy with user-defined routing to allow egress to the internet. You must ensure that cluster Operators do not access Azure APIs using a proxy; Operators must have access to Azure APIs outside of the proxy.

When using the default route table for subnets, with **0.0.0.0/0** populated automatically by Azure, all Azure API requests are routed over Azure's internal network even though the IP addresses are public. As long as the Network Security Group rules allow egress to Azure API endpoints, proxies with user-defined routing configured allow you to create private clusters with no public endpoints.

Private cluster with no internet access

You can install a private network that restricts all access to the internet, except the Azure API. This is accomplished by mirroring the release image registry locally. Your cluster must have access to the following:

- An OpenShift image registry mirror that allows for pulling container images
- Access to Azure APIs

With these requirements available, you can use user-defined routing to create private clusters with no public endpoints.

7.9.3. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.12, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

7.9.3.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICs for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

**NOTE**

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within [a region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

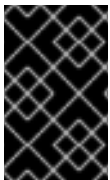
- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for.

**NOTE**

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

7.9.3.1.1. Network security group requirements

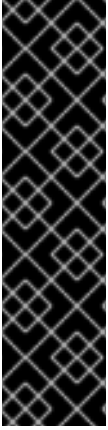
The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

**IMPORTANT**

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 7.24. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	
22623	Allows internal communication to the machine config server for provisioning machines	x	



IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Additional resources

- [About the OpenShift SDN network plugin](#)

7.9.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

7.9.3.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

7.9.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.9.5. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.9.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.9.7. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

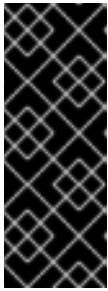
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

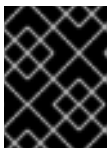
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

7.9.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.9.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.25. Required parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

7.9.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 7.26. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


7.9.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 7.27. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="border: 1px solid black; width: 60px; height: 100px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="border: 1px solid black; width: 60px; height: 100px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="488 192 592 913"></p> <p data-bbox="671 107 935 913">IMPORTANT To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="488 958 592 1155"></p> <p data-bbox="671 958 935 1155">NOTE If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	The SSH key to authenticate access to your cluster machines.  NOTE For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.	For example, sshKey: ssh-ed25519 AAAA...

7.9.7.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.



NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within a [region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.28. Additional Azure parameters

Parameter	Description	Values
compute.platform.azure.encryptionAtHost	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
compute.platform.azure.osDisk.diskSize GB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS , premium_LRS , or standardSSD_LRS . The default is premium_LRS .
compute.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	Enabled , Disabled . The default is Disabled .
compute.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
compute.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
compute.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .

Parameter	Description	Values
<code>compute.platform.azure.vmNetworkingType</code>	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
<code>compute.platform.azure.type</code>	Defines the Azure instance type for compute machines.	String
<code>compute.platform.azure.zones</code>	The availability zones where the installation program creates compute machines.	String list
<code>controlPlane.platform.azure.type</code>	Defines the Azure instance type for control plane machines.	String
<code>controlPlane.platform.azure.zones</code>	The availability zones where the installation program creates control plane machines.	String list
<code>platform.azure.defaultMachinePlatform.enableEncryptionAtHost</code>	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
<code>platform.azure.defaultMachinePlatform.ostDisk.diskEncryptionSet.name</code>	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, production_disk_encryption_set .

Parameter	Description	Values
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, production_encryption_resource_group .
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
platform.azure.defaultMachinePlatform.ossDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.ossDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.defaultMachinePlatform.zones	The availability zones where the installation program creates compute and control plane machines.	String list.
controlPlane.platform.azure.encryptionAtHost	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .

Parameter	Description	Values
controlPlane.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
controlPlane.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
controlPlane.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
controlPlane.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
controlPlane.platform.azure.vmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.region	The name of the Azure region that hosts your cluster.	Any valid region name, such as centralus .
platform.azure.zone	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example ["1", "2", "3"] .
platform.azure.defaultMachinePlatform.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
platform.azure.networkResourceGroupName	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName .	String.
platform.azure.virtualNetwork	The name of the existing VNet that you want to deploy your cluster to.	String.

Parameter	Description	Values
platform.azure.controlPlaneSubnet	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.computeSubnet	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.	Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud .
platform.azure.defaultMachinePlatform.vmmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	Accelerated or Basic . If instance type of control plane and compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .

**NOTE**

You cannot customize [Azure Availability Zones](#) or [Use tags to organize your Azure resources](#) with an Azure cluster.

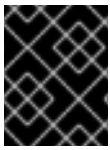
7.9.7.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 7.29. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.9.7.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 7.28. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- **r6i.***
- **t3.***
- **t3a.***

7.9.7.4. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 7.29. Machine types based on 64-bit ARM architecture

- **c6g.***
- **m6g.***

7.9.7.5. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
    osDisk:
      diskSizeGB: 1024 5
      diskType: Premium_LRS
      diskEncryptionSet:
        resourceGroup: disk_encryption_set_resource_group
        name: disk_encryption_set_name
        subscriptionId: secondary_subscription_id
      type: Standard_D8s_v3
    replicas: 3
  compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled

```

```

type: Standard_D2s_v3
encryptionAtHost: true
osDisk:
  diskSizeGB: 512 8
  diskType: Standard_LRS
  diskEncryptionSet:
    resourceGroup: disk_encryption_set_resource_group
    name: disk_encryption_set_name
    subscriptionId: secondary_subscription_id
zones: 9
- "1"
- "2"
- "3"
replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 12
    region: centralus 13
    resourceGroupName: existing_resource_group 14
    networkResourceGroupName: vnet_resource_group 15
    virtualNetwork: vnet 16
    controlPlaneSubnet: control_plane_subnet 17
    computeSubnet: compute_subnet 18
    outboundType: UserDefinedRouting 19
    cloudName: AzurePublicCloud
  pullSecret: '{"auths": ...}' 20
  fips: false 21
  sshKey: ssh-ed25519 AAAA... 22
  publish: Internal 23

```

1 10 13 20 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default,



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

- 5 8 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 9 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 Specify the name of the resource group that contains the DNS zone for your base domain.
- 14 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 15 If you use an existing VNet, specify the name of the resource group that contains it.
- 16 If you use an existing VNet, specify its name.
- 17 If you use an existing VNet, specify the name of the subnet to host the control plane machines.
- 18 If you use an existing VNet, specify the name of the subnet to host the compute machines.
- 19 You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.
- 21 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 22 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 23

How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

7.9.7.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all

destinations.

- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see [Accelerated Networking for Microsoft Azure VMs](#).

7.9.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
```

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.9.9. Finalizing user-managed encryption after installation

If you installed OpenShift Container Platform using a user-managed encryption key, you can complete the installation by creating a new storage class and granting write permissions to the Azure cluster resource group.

Procedure

1. Obtain the identity of the cluster resource group used by the installer:
 - a. If you specified an existing resource group in **install-config.yaml**, obtain its Azure identity by running the following command:

```
$ az identity list --resource-group "<existing_resource_group>"
```

- b. If you did not specify a existing resource group in **install-config.yaml**, locate the resource group that the installer created, and then obtain its Azure identity by running the following commands:

```
$ az group list
```

```
$ az identity list --resource-group "<installer_created_resource_group>"
```

2. Grant a role assignment to the cluster resource group so that it can write to the Disk Encryption Set by running the following command:

```
$ az role assignment create --role "<privileged_role>" \
  --assignee "<resource_group_identity>"
```

1 Specifies an Azure role that has read/write permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.

2 Specifies the identity of the cluster resource group.

- Obtain the **id** of the disk encryption set you created prior to installation by running the following command:

```
$ az disk-encryption-set show -n <disk_encryption_set_name> \1
--resource-group <resource_group_name> \2
```

- Specifies the name of the disk encryption set.
- Specifies the resource group that contains the disk encryption set. The **id** is in the format of **"/subscriptions/.../resourceGroups/.../providers/Microsoft.Compute/diskEncryptionSets/..."**.

- Obtain the identity of the cluster service principal by running the following command:

```
$ az identity show -g <cluster_resource_group> \1
-n <cluster_service_principal_name> \2
--query principalId --out tsv
```

- Specifies the name of the cluster resource group created by the installation program.
- Specifies the name of the cluster service principal created by the installation program. The identity is in the format of **12345678-1234-1234-1234-1234567890**.

- Create a role assignment that grants the cluster service principal necessary privileges to the disk encryption set by running the following command:

```
$ az role assignment create --assignee <cluster_service_principal_id> \1
--role <privileged_role> \2
--scope <disk_encryption_set_id> \3
```

- Specifies the ID of the cluster service principal obtained in the previous step.
- Specifies the Azure role name. You can use the **Contributor** role or a custom role with the necessary permissions.
- Specifies the ID of the disk encryption set.

- Create a storage class that uses the user-managed disk encryption set:

- Save the following storage class definition to a file, for example **storage-class-definition.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: managed-premium
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
  kind: Managed
  diskEncryptionSetID: "<disk_encryption_set_ID>" \1
  resourceGroup: "<resource_group_name>" \2
```

```
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: WaitForFirstConsumer
```

- 1 Specifies the ID of the disk encryption set that you created in the prerequisite steps, for example `"/subscriptions/xxxxxx-xxxxx-xxxxx/resourceGroups/test-encryption/providers/Microsoft.Compute/diskEncryptionSets/disk-encryption-set-xxxxxx"`.
- 2 Specifies the name of the resource group used by the installer. This is the same resource group from the first step.

- b. Create the storage class **managed-premium** from the file you created by running the following command:

```
$ oc create -f storage-class-definition.yaml
```

7. Select the **managed-premium** storage class when you create persistent volumes to use encrypted storage.

7.9.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

7.9.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.9.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.9.13. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

7.10. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION

In OpenShift Container Platform version 4.12, you can install a cluster on Microsoft Azure into a government region. To configure the government region, you modify parameters in the **install-config.yaml** file before you install the cluster.

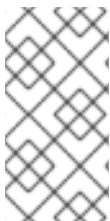
7.10.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster and determined the tested and validated government region to deploy the cluster to.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .
- If you use customer-managed encryption keys, you [prepared your Azure environment for encryption](#).

7.10.2. Azure government regions

OpenShift Container Platform supports deploying a cluster to [Microsoft Azure Government \(MAG\)](#) regions. MAG is specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads on Azure. MAG is composed of government-only data center regions, all granted an [Impact Level 5 Provisional Authorization](#).

Installing to a MAG region requires manually configuring the Azure Government dedicated cloud instance and region in the **install-config.yaml** file. You must also update your service principal to reference the appropriate government environment.



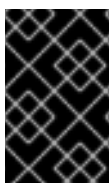
NOTE

The Azure government region cannot be selected using the guided terminal prompts from the installation program. You must define the region manually in the **install-config.yaml** file. Remember to also set the dedicated cloud instance, like **AzureUSGovernmentCloud**, based on the region specified.

7.10.3. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.
 - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

7.10.3.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder to resolve the cluster's private DNS records. The cluster's machines use **168.63.129.16** internally for DNS resolution. For more information, see [What is Azure Private DNS?](#) and [What is IP address 168.63.129.16?](#) in the Azure documentation.

The cluster still requires access to internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A **BaseDomainResourceGroup**, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

7.10.3.1.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

7.10.3.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the **install-config.yaml** file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.

You must ensure the following items are available before setting user-defined routing:

- Egress to the internet is possible to pull container images, unless using an OpenShift image registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the *Configuring your firewall* section.

There are several pre-existing networking setups that are supported for internet access using user-defined routing.

Private cluster with network address translation

You can use [Azure VNET network address translation \(NAT\)](#) to provide outbound internet access for the subnets in your cluster. You can reference [Create a NAT gateway using Azure CLI](#) in the Azure documentation for configuration instructions.

When using a VNet setup with Azure NAT and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with Azure Firewall

You can use Azure Firewall to provide outbound routing for the VNet used to install the cluster. You can learn more about [providing user-defined routing with Azure Firewall](#) in the Azure documentation.

When using a VNet setup with Azure Firewall and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with a proxy configuration

You can use a proxy with user-defined routing to allow egress to the internet. You must ensure that cluster Operators do not access Azure APIs using a proxy; Operators must have access to Azure APIs outside of the proxy.

When using the default route table for subnets, with **0.0.0.0/0** populated automatically by Azure, all Azure API requests are routed over Azure's internal network even though the IP addresses are public. As long as the Network Security Group rules allow egress to Azure API endpoints, proxies with user-defined routing configured allow you to create private clusters with no public endpoints.

Private cluster with no internet access

You can install a private network that restricts all access to the internet, except the Azure API. This is accomplished by mirroring the release image registry locally. Your cluster must have access to the following:

- An OpenShift image registry mirror that allows for pulling container images
- Access to Azure APIs

With these requirements available, you can use user-defined routing to create private clusters with no public endpoints.

7.10.4. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.12, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

7.10.4.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

**NOTE**

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

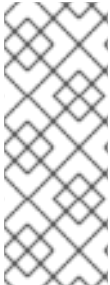
If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

**NOTE**

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within [a region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.

**NOTE**

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

7.10.4.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.



IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 7.30. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	
22623	Allows internal communication to the machine config server for provisioning machines	x	



IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Additional resources

- [About the OpenShift SDN network plugin](#)

7.10.4.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

7.10.4.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

7.10.5. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

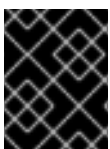
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.10.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

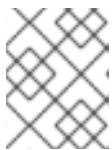
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.10.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

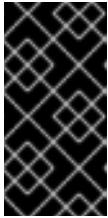
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.10.8. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

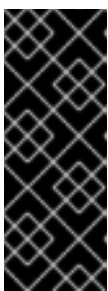
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

7.10.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.10.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.31. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> , <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

7.10.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 7.32. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

7.10.8.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:

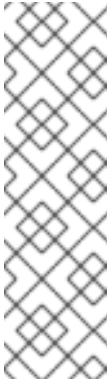

Table 7.33. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String


Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 . Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see <i>Supported installation methods for different platforms</i> in <i>Selecting a cluster installation method and preparing it for users</i> .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

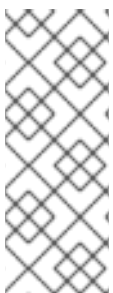
Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 593 1393" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 593 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true

Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	The SSH key to authenticate access to your cluster machines.  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

7.10.8.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.



NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across [these availability zones](#) within [a region](#). To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.34. Additional Azure parameters

Parameter	Description	Values
compute.platform.azure.encryptionAtHost	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
compute.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS , premium_LRS , or standardSSD_LRS . The default is premium_LRS .
compute.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	Enabled , Disabled . The default is Disabled .
compute.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
compute.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
compute.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .

Parameter	Description	Values
<code>compute.platform.azure.vmNetworkingType</code>	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .
<code>compute.platform.azure.type</code>	Defines the Azure instance type for compute machines.	String
<code>compute.platform.azure.zones</code>	The availability zones where the installation program creates compute machines.	String list
<code>controlPlane.platform.azure.type</code>	Defines the Azure instance type for control plane machines.	String
<code>controlPlane.platform.azure.zones</code>	The availability zones where the installation program creates control plane machines.	String list
<code>platform.azure.defaultMachinePlatform.enableEncryptionAtHost</code>	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .
<code>platform.azure.defaultMachinePlatform.ostDisk.diskEncryptionSet.name</code>	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, production_disk_encryption_set .

Parameter	Description	Values
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, production_encryption_resource_group .
platform.azure.defaultMachinePlatform.ossDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
platform.azure.defaultMachinePlatform.ossDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.ossDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.defaultMachinePlatform.zones	The availability zones where the installation program creates compute and control plane machines.	String list.
controlPlane.platform.azure.encryptedHost	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	true or false . The default is false .

Parameter	Description	Values
controlPlane.platform.azure.osDisk.diskEncryptionSet.resourceGroup	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example production_encryption_resource_group .
controlPlane.platform.azure.osDisk.diskEncryptionSet.name	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example production_disk_encryption_set .
controlPlane.platform.azure.osDisk.diskEncryptionSet.subscriptionId	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format 00000000-0000-0000-0000-000000000000 .
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS or standardSSD_LRS . The default is premium_LRS .
controlPlane.platform.azure.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .
controlPlane.platform.azure.vmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .	Accelerated or Basic .

Parameter	Description	Values
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.region	The name of the Azure region that hosts your cluster.	Any valid region name, such as centralus .
platform.azure.zone	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example ["1", "2", "3"] .
platform.azure.defaultMachinePlatform.ultraSSDCapability	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	Enabled, Disabled . The default is Disabled .

Parameter	Description	Values
platform.azure.networkResourceGroupName	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName .	String.
platform.azure.virtualNetwork	The name of the existing VNet that you want to deploy your cluster to.	String.
platform.azure.controlPlaneSubnet	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.computeSubnet	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example 10.0.0.0/16 .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.	Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud .
platform.azure.defaultMachinePlatform.vmmNetworkingType	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	Accelerated or Basic . If instance type of control plane and compute machines support Accelerated networking, by default, the installer enables Accelerated networking, otherwise the default networking type is Basic .

**NOTE**

You cannot customize [Azure Availability Zones](#) or [Use tags to organize your Azure resources](#) with an Azure cluster.

7.10.8.2. Minimum resource requirements for cluster installation

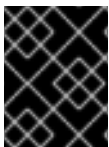
Each cluster machine must meet the following minimum requirements:

Table 7.35. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.10.8.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

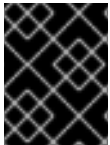
Example 7.30. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***

- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

7.10.8.4. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com ①
controlPlane: ②
  hyperthreading: Enabled ③ ④
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 ⑤
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
        type: Standard_D8s_v3
      replicas: 3
    compute: ⑥
  - hyperthreading: Enabled ⑦
    name: worker
    platform:
      azure:

```

```

ultraSSDCapability: Enabled
type: Standard_D2s_v3
encryptionAtHost: true
osDisk:
  diskSizeGB: 512 8
  diskType: Standard_LRS
  diskEncryptionSet:
    resourceGroup: disk_encryption_set_resource_group
    name: disk_encryption_set_name
    subscriptionId: secondary_subscription_id
zones: 9
- "1"
- "2"
- "3"
replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 12
    region: usgovvirginia
    resourceGroupName: existing_resource_group 13
    networkResourceGroupName: vnet_resource_group 14
    virtualNetwork: vnet 15
    controlPlaneSubnet: control_plane_subnet 16
    computeSubnet: compute_subnet 17
    outboundType: UserDefinedRouting 18
    cloudName: AzureUSGovernmentCloud 19
  pullSecret: '{"auths": ...}' 20
  fips: false 21
  sshKey: ssh-ed25519 AAAA... 22
  publish: Internal 23

```

1 10 20 Required.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

- 4 Whether to enable or disable simultaneous multithreading, or **hypertreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

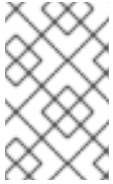
- 5 8 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 9 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 Specify the name of the resource group that contains the DNS zone for your base domain.
- 13 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 14 If you use an existing VNet, specify the name of the resource group that contains it.
- 15 If you use an existing VNet, specify its name.
- 16 If you use an existing VNet, specify the name of the subnet to host the control plane machines.
- 17 If you use an existing VNet, specify the name of the subnet to host the compute machines.
- 18 You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.
- 19 Specify the name of the Azure cloud environment to deploy your cluster to. Set **AzureUSGovernmentCloud** to deploy to a Microsoft Azure Government (MAG) region. The default value is **AzurePublicCloud**.
- 21 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 22 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 23** How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

7.10.8.5. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1** A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see [Accelerated Networking for Microsoft Azure VMs](#).

7.10.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

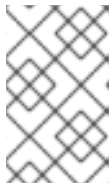
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
```

```
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.10.10. Finalizing user-managed encryption after installation

If you installed OpenShift Container Platform using a user-managed encryption key, you can complete the installation by creating a new storage class and granting write permissions to the Azure cluster resource group.

Procedure

1. Obtain the identity of the cluster resource group used by the installer:
 - a. If you specified an existing resource group in **install-config.yaml**, obtain its Azure identity by running the following command:

```
$ az identity list --resource-group "<existing_resource_group>"
```

- b. If you did not specify a existing resource group in **install-config.yaml**, locate the resource group that the installer created, and then obtain its Azure identity by running the following commands:

```
$ az group list
```

```
$ az identity list --resource-group "<installer_created_resource_group>"
```

2. Grant a role assignment to the cluster resource group so that it can write to the Disk Encryption Set by running the following command:

```
$ az role assignment create --role "<privileged_role>" \
  --assignee "<resource_group_identity>"
```


- 1 Specifies an Azure role that has read/write permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.
 - 2 Specifies the identity of the cluster resource group.
3. Obtain the **id** of the disk encryption set you created prior to installation by running the following command:

```
$ az disk-encryption-set show -n <disk_encryption_set_name> \ 1
--resource-group <resource_group_name> 2
```

- 1 Specifies the name of the disk encryption set.
- 2 Specifies the resource group that contains the disk encryption set. The **id** is in the format of **"/subscriptions/.../resourceGroups/.../providers/Microsoft.Compute/diskEncryptionSets/..."**.

4. Obtain the identity of the cluster service principal by running the following command:

```
$ az identity show -g <cluster_resource_group> \ 1
-n <cluster_service_principal_name> \ 2
--query principalId --out tsv
```

- 1 Specifies the name of the cluster resource group created by the installation program.
- 2 Specifies the name of the cluster service principal created by the installation program. The identity is in the format of **12345678-1234-1234-1234-1234567890**.

5. Create a role assignment that grants the cluster service principal **Contributor** privileges to the disk encryption set by running the following command:

```
$ az role assignment create --assignee <cluster_service_principal_id> \ 1
--role 'Contributor' \ /
--scope <disk_encryption_set_id> \ 2
```

- 1 Specifies the ID of the cluster service principal obtained in the previous step.
- 2 Specifies the ID of the disk encryption set.

6. Create a storage class that uses the user-managed disk encryption set:

- a. Save the following storage class definition to a file, for example **storage-class-definition.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: managed-premium
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
```

```
kind: Managed
diskEncryptionSetID: "<disk_encryption_set_ID>" 1
resourceGroup: "<resource_group_name>" 2
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: WaitForFirstConsumer
```

- 1** Specifies the ID of the disk encryption set that you created in the prerequisite steps, for example `"/subscriptions/xxxxxx-xxxxx-xxxxx/resourceGroups/test-encryption/providers/Microsoft.Compute/diskEncryptionSets/disk-encryption-set-xxxxxx"`.
- 2** Specifies the name of the resource group used by the installer. This is the same resource group from the first step.

- b. Create the storage class **managed-premium** from the file you created by running the following command:

```
$ oc create -f storage-class-definition.yaml
```

7. Select the **managed-premium** storage class when you create persistent volumes to use encrypted storage.

7.10.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

7.10.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

7.10.13. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct,

either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

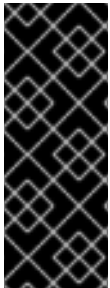
7.10.14. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

7.11. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES

In OpenShift Container Platform version 4.12, you can install a cluster on Microsoft Azure by using infrastructure that you provide.

Several [Azure Resource Manager](#) (ARM) templates are provided to assist in completing these steps or to help model your own.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several ARM templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

7.11.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure account](#) to host the cluster.
- You downloaded the Azure CLI and installed it on your computer. See [Install the Azure CLI](#) in the Azure documentation. The documentation below was last tested using version **2.38.0** of the Azure CLI. Azure CLI commands might perform differently based on the version you use.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

7.11.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

7.11.3. Configuring your Azure project

Before you can install OpenShift Container Platform, you must configure an Azure project to host it.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

7.11.3.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default [Azure subscription and service limits, quotas, and constraints](#) affect your ability to install OpenShift Container Platform clusters.

**IMPORTANT**


Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Default Azure limit	Description
vCPU	40	20 per region	<p>A default cluster requires 40 vCPUs, so you must increase the account limit.</p> <p>By default, each cluster creates the following instances:</p> <ul style="list-style-type: none"> • One bootstrap machine, which is removed after installation • Three control plane machines • Three compute machines <p>Because the bootstrap machine uses Standard_D4s_v3 machines, which use 4 vCPUs, the control plane machines use Standard_D8s_v3 virtual machines, which use 8 vCPUs, and the worker machines use Standard_D4s_v3 virtual machines, which use 4 vCPUs, a default cluster requires 40 vCPUs. The bootstrap node VM, which uses 4 vCPUs, is used only during installation.</p> <p>To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.</p>
OS Disk	7		<p>Each cluster machine must have a minimum of 100 GB of storage and 300 IOPS. While these are the minimum supported values, faster storage is recommended for production clusters and clusters with intensive workloads. For more information about optimizing storage for performance, see the page titled "Optimizing storage" in the "Scalability and performance" section.</p>
VNet	1	1000 per region	<p>Each default cluster requires one Virtual Network (VNet), which contains two subnets.</p>
Network interfaces	7	65,536 per region	<p>Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.</p>

Component	Number of components required by default	Default Azure limit	Description						
Network security groups	2	5000	<p>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</p> <table border="1"> <tr> <td>control plane</td> <td>Allows the control plane machines to be reached on port 6443 from anywhere</td> </tr> <tr> <td>node</td> <td>Allows worker nodes to be reached from the internet on ports 80 and 443</td> </tr> </table>	control plane	Allows the control plane machines to be reached on port 6443 from anywhere	node	Allows worker nodes to be reached from the internet on ports 80 and 443		
control plane	Allows the control plane machines to be reached on port 6443 from anywhere								
node	Allows worker nodes to be reached from the internet on ports 80 and 443								
Network load balancers	3	1000 per region	<p>Each cluster creates the following load balancers:</p> <table border="1"> <tr> <td>default</td> <td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td> </tr> <tr> <td>internal</td> <td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td> </tr> <tr> <td>external</td> <td>Public IP address that load balances requests to port 6443 across control plane machines</td> </tr> </table> <p>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</p>	default	Public IP address that load balances requests to ports 80 and 443 across worker machines	internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines	external	Public IP address that load balances requests to port 6443 across control plane machines
default	Public IP address that load balances requests to ports 80 and 443 across worker machines								
internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines								
external	Public IP address that load balances requests to port 6443 across control plane machines								
Public IP addresses	3		Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.						
Private IP addresses	7		The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.						

Component	Number of components required by default	Default Azure limit	Description
Spot VM vCPUs (optional)	0 If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node.	20 per region	<p>This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>Using spot VMs for control plane nodes is not recommended.</p> </div> </div>

Additional resources

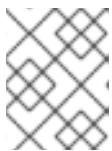
- [Optimizing storage](#)

7.11.3.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.



NOTE

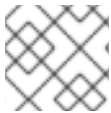
For more information about purchasing domains through Azure, see [Buy a custom domain name for Azure App Service](#) in the Azure documentation.

2. If you are using an existing domain and registrar, migrate its DNS to Azure. See [Migrate an active DNS name to Azure App Service](#) in the Azure documentation.
3. Configure DNS for your domain. Follow the steps in the [Tutorial: Host your domain in Azure DNS](#) in the Azure documentation to create a public hosted zone for your domain or subdomain, extract the new authoritative name servers, and update the registrar records for the name servers that your domain uses.
Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
4. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain.

You can view Azure's DNS solution by visiting this [example for creating DNS zones](#).

7.11.3.3. Increasing Azure account limits

To increase an account limit, file a support request on the Azure portal.

**NOTE**

You can increase only one type of quota per support request.

Procedure

1. From the Azure portal, click **Help + support** in the lower left corner.
2. Click **New support request** and then select the required values:
 - a. From the **Issue type** list, select **Service and subscription limits (quotas)**
 - b. From the **Subscription** list, select the subscription to modify.
 - c. From the **Quota type** list, select the quota to increase. For example, select **Compute-VM (cores-vCPUs) subscription limit increases** to increase the number of vCPUs, which is required to install a cluster.
 - d. Click **Next: Solutions**.
3. On the **Problem Details** page, provide the required information for your quota increase:
 - a. Click **Provide details** and provide the required details in the **Quota details** window.
 - b. In the SUPPORT METHOD and CONTACT INFO sections, provide the issue severity and your contact details.
4. Click **Next: Review + create** and then click **Create**.

7.11.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

7.11.3.5. Required Azure roles

OpenShift Container Platform needs a service principal so it can manage Microsoft Azure resources. Before you can create a service principal, review the following information:

Your Azure account subscription must have the following roles:

- **User Access Administrator**
- **Contributor**

Your Azure Active Directory (AD) must have the following permission:

- **"microsoft.directory/servicePrincipals/createAsOwner"**

To set roles on the Azure portal, see the [Manage access to Azure resources using RBAC and the Azure portal](#) in the Azure documentation.

7.11.3.6. Required Azure permissions for user-provisioned infrastructure

When you assign **Contributor** and **User Access Administrator** roles to the service principal, you automatically grant all the required permissions.

If your organization's security policies require a more restrictive set of permissions, you can create a [custom role](#) with the necessary permissions. The following permissions are required for creating an OpenShift Container Platform cluster on Microsoft Azure.

Example 7.31. Required permissions for creating authorization resources

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

Example 7.32. Required permissions for creating compute resources

- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**
- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**

- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**
- **Microsoft.Compute/virtualMachines/deallocate/action**

Example 7.33. Required permissions for creating identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

Example 7.34. Required permissions for creating network resources

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**

- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

Example 7.35. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 7.36. Required permissions for creating a resource group

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

Example 7.37. Required permissions for creating resource tags

- **Microsoft.Resources/tags/write**

Example 7.38. Required permissions for creating storage resources

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

Example 7.39. Required permissions for creating deployments

- **Microsoft.Resources/deployments/read**
- **Microsoft.Resources/deployments/write**
- **Microsoft.Resources/deployments/validate/action**
- **Microsoft.Resources/deployments/operationstatuses/read**

Example 7.40. Optional permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/availabilitySets/write**

Example 7.41. Optional permissions for creating marketplace virtual machine resources

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

Example 7.42. Optional permissions for enabling user-managed encryption

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**

- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

The following permissions are required for deleting an OpenShift Container Platform cluster on Microsoft Azure.

Example 7.43. Required permissions for deleting authorization resources

- **Microsoft.Authorization/roleAssignments/delete**

Example 7.44. Required permissions for deleting compute resources

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/images/delete**

Example 7.45. Required permissions for deleting identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

Example 7.46. Required permissions for deleting network resources

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**

- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**

Example 7.47. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 7.48. Required permissions for deleting a resource group

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

Example 7.49. Required permissions for deleting storage resources

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**



NOTE

To install OpenShift Container Platform on Azure, you must scope the permissions related to resource group creation to your subscription. After the resource group is created, you can scope the rest of the permissions to the created resource group. If the public DNS zone is present in a different resource group, then the network DNS zone related permissions must always be applied to your subscription.

You can scope all the permissions to your subscription when deleting an OpenShift Container Platform cluster.

7.11.3.7. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the [Azure CLI](#).
- Your Azure account has the required roles for the subscription that you use.
- If you want to use a custom role, you have created a [custom role](#) with the required permissions listed in the *Required Azure permissions for user-provisioned infrastructure* section.

Procedure

1. Log in to the Azure CLI:

```
$ az login
```

2. If your Azure account uses subscriptions, ensure that you are using the right subscription:

- a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": "AzureCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

- b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", 1
  "user": {
```

```

    "name": "you@example.com",
    "type": "user"
  }
}

```

- 1 Ensure that the value of the **tenantId** parameter is the correct subscription ID.

- c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> 1
```

- 1 Specify the subscription ID.

- d. Verify the subscription ID update:

```
$ az account show
```

Example output

```

{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}

```

3. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.
4. Create the service principal for your account:

```

$ az ad sp create-for-rbac --role <role_name> \ 1
  --name <service_principal> \ 2
  --scopes /subscriptions/<subscription_id> 3

```

- 1 Defines the role name. You can use the **Contributor** role, or you can specify a custom role which contains the necessary permissions.
- 2 Defines the service principal name.
- 3 Specifies the subscription ID.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
```

The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the credentials into your source control. For more information, see <https://aka.ms/azadsp-cli>

```
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>,
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

- Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.
- If you applied the **Contributor** role to your service principal, assign the **User Administrator Access** role by running the following command:

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
```

- Specify the **appId** parameter value for your service principal.

Additional resources

- For more information about CCO modes, see [About the Cloud Credential Operator](#).

7.11.3.8. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription.

Supported Azure public regions

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)

- **germanywestcentral** (Germany West Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **mexicocentral** (Mexico Central)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

Supported Azure Government regions

Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the [Azure documentation](#). Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

7.11.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

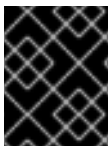
This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

7.11.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 7.36. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

7.11.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 7.37. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.



IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

7.11.4.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 7.50. Machine types based on 64-bit x86 architecture

- **c4.***
- **c5.***
- **c5a.***

- i3.*
- m4.*
- m5.*
- m5a.*
- m6a.*
- m6i.*
- r4.*
- r5.*
- r5a.*
- r6i.*
- t3.*
- t3a.*

7.11.5. Selecting an Azure Marketplace image

If you are deploying an OpenShift Container Platform cluster using the Azure Marketplace offering, you must first obtain the Azure Marketplace image. The installation program uses this image to deploy worker nodes. When obtaining your image, consider the following:

- While the images are the same, the Azure Marketplace publisher is different depending on your region. If you are located in North America, specify **redhat** as the publisher. If you are located in EMEA, specify **redhat-limited** as the publisher.
- The offer includes a **rh-ocp-worker** SKU and a **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker** SKU represents a Hyper-V generation version 2 VM image. The default instance types used in OpenShift Container Platform are version 2 compatible. If you plan to use an instance type that is only version 1 compatible, use the image associated with the **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker-gen1** SKU represents a Hyper-V version 1 VM image.



IMPORTANT

Installing images with the Azure marketplace is not supported on clusters with 64-bit ARM instances.

Prerequisites

- You have installed the Azure CLI client (**az**).
- Your Azure account is entitled for the offer and you have logged into this account with the Azure CLI client.

Procedure

1. Display all of the available OpenShift Container Platform images by running one of the following commands:

- North America:

```
$ az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
```

Example output

Offer	Publisher	Skus	Urn	Version
rh-ocp-worker-ocpworker:4.8.2021122100	RedHat	rh-ocp-worker	RedHat:rh-ocp-worker:rh-ocp-worker:4.8.2021122100	
rh-ocp-worker-gen1:4.8.2021122100	RedHat	rh-ocp-worker-gen1	RedHat:rh-ocp-worker:rh-ocp-worker-gen1:4.8.2021122100	

- EMEA:

```
$ az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

Example output

Offer	Publisher	Skus	Urn	Version
rh-ocp-worker-worker:4.8.2021122100	redhat-limited	rh-ocp-worker	redhat-limited:rh-ocp-worker:rh-ocp-worker:4.8.2021122100	
rh-ocp-worker-gen1:4.8.2021122100	redhat-limited	rh-ocp-worker-gen1	redhat-limited:rh-ocp-worker:rh-ocp-worker-gen1:4.8.2021122100	



NOTE

Regardless of the version of OpenShift Container Platform that you install, the correct version of the Azure Marketplace image to use is 4.8. If required, your VMs are automatically upgraded as part of the installation process.

2. Inspect the image for your offer by running one of the following commands:

- North America:

```
$ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. Review the terms of the offer by running one of the following commands:

- North America:

```
$ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```


- EMEA:

```
$ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

4. Accept the terms of the offering by running one of the following commands:

- North America:

```
$ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

5. Record the image details of your offer. If you use the Azure Resource Manager (ARM) template to deploy your worker nodes:

- Update **storageProfile.imageReference** by deleting the **id** parameter and adding the **offer**, **publisher**, **sku**, and **version** parameters by using the values from your offer.
- Specify a **plan** for the virtual machines (VMs).

Example 06_workers.json ARM template with an updated storageProfile.imageReference object and a specified plan

```
...
  "plan" : {
    "name": "rh-ocp-worker",
    "product": "rh-ocp-worker",
    "publisher": "redhat"
  },
  "dependsOn" : [
    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
  ],
  "properties" : {
...
  "storageProfile": {
    "imageReference": {
      "offer": "rh-ocp-worker",
      "publisher": "redhat",
      "sku": "rh-ocp-worker",
      "version": "4.8.2021122100"
    }
...
  }
...
}
```

7.11.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

7.11.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

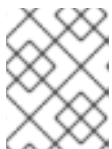
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

7.11.8. Creating the installation files for Azure

To install OpenShift Container Platform on Microsoft Azure using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

7.11.8.1. Optional: Creating a separate /var partition

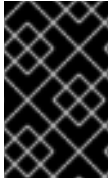
It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
```

```

name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> ❶
      partitions:
        - label: var
          start_mib: <partition_start_offset> ❷
          size_mib: <partition_size> ❸
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] ❹
          with_mount_unit: true

```

- ❶ The storage device name of the disk that you want to partition.
- ❷ When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- ❸ The size of the data partition in mebibytes.
- ❹ The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

7.11.8.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **azure** as the platform to target.
- iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:
 - **azure subscription id** The subscription ID to use for the cluster. Specify the **id** value in your account output.
 - **azure tenant id** The tenant ID. Specify the **tenantId** value in your account output.
 - **azure service principal client id** The value of the **appId** parameter for the service principal.

- **azure service principal client secret** The value of the **password** parameter for the service principal.
- iv. Select the region to deploy the cluster to.
 - v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.
 - vi. Enter a descriptive name for your cluster.



IMPORTANT

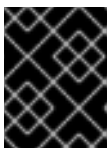
All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

- vii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
- c. Optional: If you do not want the cluster to provision compute machines, empty the compute pool by editing the resulting **install-config.yaml** file to set **replicas** to **0** for the **compute** pool:

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

1 Set to **0**.

2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

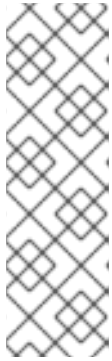
7.11.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to

bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

7.11.8.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure.

**NOTE**

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Export common variables found in the **install-config.yaml** to be used by the provided ARM templates:

```
$ export CLUSTER_NAME=<cluster_name> 1
$ export AZURE_REGION=<azure_region> 2
$ export SSH_KEY=<ssh_key> 3
$ export BASE_DOMAIN=<base_domain> 4
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> 5
```

- 1 The value of the **.metadata.name** attribute from the **install-config.yaml** file.
- 2 The region to deploy the cluster into, for example **centralus**. This is the value of the **.platform.azure.region** attribute from the **install-config.yaml** file.

- 3 The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the **.sshKey** attribute from the **install-config.yaml**
- 4 The base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster. This is the value of the **.baseDomain** attribute from the **install-config.yaml** file.
- 5 The resource group where the public DNS zone exists. This is the value of the **.platform.azure.baseDomainResourceGroupName** attribute from the **install-config.yaml** file.

For example:

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. Export the kubeadmin credentials:

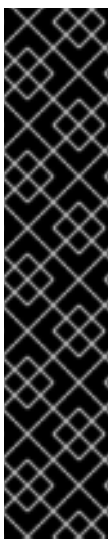
```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

7.11.8.5. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

- a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
- b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
- c. Save and exit the file.

5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
  id: mycluster-100419-private-zone
```

```
publicZone: 2
  id: example.openshift.com
  status: {}
```

- 1 2 Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

- a. Export the infrastructure ID by using the following command:

```
$ export INFRA_ID=<infra_id> 1
```

- 1 The OpenShift Container Platform cluster has been assigned an identifier (**INFRA_ID**) in the form of **<cluster_name>-<random_string>**. This will be used as the base name for most resources created using the provided ARM templates. This is the value of the **.status.infrastructureName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

- b. Export the resource group by using the following command:

```
$ export RESOURCE_GROUP=<resource_group> 1
```

- 1 All resources created in this Azure deployment exists as part of a [resource group](#). The resource group name is also based on the **INFRA_ID**, in the form of **<cluster_name>-<random_string>-rg**. This is the value of the **.status.platformStatus.azure.resourceGroupName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

7. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$. /openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

7.11.9. Creating the Azure resource group

You must create a Microsoft Azure [resource group](#) and an identity for that resource group. These are both used during the installation of your OpenShift Container Platform cluster on Azure.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Create the resource group in a supported Azure region:

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

2. Create an Azure identity for the resource group:

```
$ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
```

This is used to grant the required access to Operators in your cluster. For example, this allows the Ingress Operator to create a public IP and its load balancer. You must assign the Azure identity to a role.

3. Grant the Contributor role to the Azure identity:

- a. Export the following variables required by the Azure role assignment:

```
$ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
```

```
$ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`
```

- b. Assign the Contributor role to the identity:

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"
```



NOTE

If you want to assign a custom role with all the required permissions to the identity, run the following command:

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role <custom_role> \ 1 --scope "${RESOURCE_GROUP_ID}"
```

- 1** Specifies the custom role name.

7.11.10. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally. You must copy and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a storage container so they are accessible during deployment.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```



WARNING

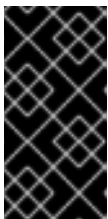
The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your **CLUSTER_NAME** variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see [Resolve errors for storage account names](#) in the Azure documentation.

2. Export the storage account key as an environment variable:

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. Export the URL of the RHCOS VHD to an environment variable:

```
$ export VHD_URL=`openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64."rhel-coreos-extensions"."azure-disk".url`
```



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Create the storage container for the VHD:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-
key ${ACCOUNT_KEY}
```

5. Copy the local VHD to a blob:

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri
"${VHD_URL}"
```

6. Create a blob storage container and upload the generated **bootstrap.ign** file:

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

7.11.11. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, [Azure's DNS solution](#) is used, so you will create a new public DNS zone for external (internet) visibility and a private DNS zone for internal cluster resolution.



NOTE

The public DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the public DNS zone; be sure the installation config you generated earlier reflects that scenario.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Create the new public DNS zone in the resource group exported in the **BASE_DOMAIN_RESOURCE_GROUP** environment variable:

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can skip this step if you are using a public DNS zone that already exists.

2. Create the private DNS zone in the same resource group as the rest of this deployment:

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can learn more about [configuring a public DNS zone in Azure](#) by visiting that section.

7.11.12. Creating a VNet in Azure

You must create a virtual network (VNet) in Microsoft Azure for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.



NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the **ARM template for the VNet** section of this topic and save it as **01_vnet.json** in your cluster's installation directory. This template describes the VNet that your cluster requires.
2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

- 1** The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Link the VNet template to the private DNS zone:

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
  ${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
  -e false
```

7.11.12.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

Example 7.51. 01_vnet.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
```

```

    "minLength" : 1,
    "metadata" : {
      "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "addressPrefix" : "10.0.0.0/16",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetPrefix" : "10.0.0.0/24",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetPrefix" : "10.0.1.0/24",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/virtualNetworks",
      "name" : "[variables('virtualNetworkName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
      ],
      "properties" : {
        "addressSpace" : {
          "addressPrefixes" : [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets" : [
          {
            "name" : "[variables('masterSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('masterSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
                "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
              }
            }
          },
          {
            "name" : "[variables('nodeSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('nodeSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
                "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
              }
            }
          }
        ]
      }
    }
  ]
}

```

```

    },
    {
      "type" : "Microsoft.Network/networkSecurityGroups",
      "name" : "[variables('clusterNsgName')]",
      "apiVersion" : "2018-10-01",
      "location" : "[variables('location')]",
      "properties" : {
        "securityRules" : [
          {
            "name" : "apiserver_in",
            "properties" : {
              "protocol" : "Tcp",
              "sourcePortRange" : "*",
              "destinationPortRange" : "6443",
              "sourceAddressPrefix" : "*",
              "destinationAddressPrefix" : "*",
              "access" : "Allow",
              "priority" : 101,
              "direction" : "Inbound"
            }
          }
        ]
      }
    }
  ]
}

```

7.11.13. Deploying the RHCOS cluster image for the Azure infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure for your OpenShift Container Platform nodes.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.
- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the **ARM template for image storage** section of this topic and save it as **02_storage.json** in your cluster's installation directory. This template describes the image storage that your cluster requires.
2. Export the RHCOS VHD blob URL as a variable:

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. Deploy the cluster image:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" 1 \
  --parameters baseName="${INFRA_ID}" 2
```

- 1** The blob URL of the RHCOS VHD to be used to create master and worker machines.
- 2** The base name to be used in resource names; this is usually the cluster's infrastructure ID.

7.11.13.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

Example 7.52. 02_storage.json ARM template

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "architecture": {
      "type": "string",
      "metadata": {
        "description": "The architecture of the Virtual Machines"
      },
      "defaultValue": "x64",
      "allowedValues": [
        "Arm64",
        "x64"
      ]
    },
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "storageAccount": {
      "type": "string",
      "metadata": {
        "description": "The Storage Account name"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and worker machines is located"
      }
    }
  },
  "variables": {
```

```

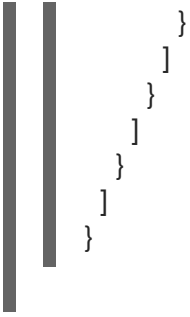
"location": "[resourceGroup().location]",
"galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
"imageName": "[parameters('baseName')]",
"imageNameGen2": "[concat(parameters('baseName'), '-gen2')]",
"imageRelease": "1.0.0"
},
"resources": [
{
  "apiVersion": "2021-10-01",
  "type": "Microsoft.Compute/galleries",
  "name": "[variables('galleryName')]",
  "location": "[variables('location')]",
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "images",
      "name": "[variables('imageName')]",
      "location": "[variables('location')]",
      "dependsOn": [
        "[variables('galleryName')]"
      ],
      "properties": {
        "architecture": "[parameters('architecture')]",
        "hyperVGeneration": "V1",
        "identifier": {
          "offer": "rhcos",
          "publisher": "RedHat",
          "sku": "basic"
        },
        "osState": "Generalized",
        "osType": "Linux"
      },
    },
  ],
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "versions",
      "name": "[variables('imageRelease')]",
      "location": "[variables('location')]",
      "dependsOn": [
        "[variables('imageName')]"
      ],
      "properties": {
        "publishingProfile": {
          "storageAccountType": "Standard_LRS",
          "targetRegions": [
            {
              "name": "[variables('location')]",
              "regionalReplicaCount": "1"
            }
          ]
        },
        "storageProfile": {
          "osDiskImage": {
            "source": {
              "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",

```

```

        "uri": "[parameters('vhdBlobURL')]"
      }
    }
  }
}
]
},
{
  "apiVersion": "2021-10-01",
  "type": "images",
  "name": "[variables('imageNameGen2')]",
  "location": "[variables('location')]",
  "dependsOn": [
    "[variables('galleryName')]"
  ],
  "properties": {
    "architecture": "[parameters('architecture')]",
    "hyperVGeneration": "V2",
    "identifier": {
      "offer": "rhcos-gen2",
      "publisher": "RedHat-gen2",
      "sku": "gen2"
    },
    "osState": "Generalized",
    "osType": "Linux"
  },
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "versions",
      "name": "[variables('imageRelease')]",
      "location": "[variables('location')]",
      "dependsOn": [
        "[variables('imageNameGen2')]"
      ],
      "properties": {
        "publishingProfile": {
          "storageAccountType": "Standard_LRS",
          "targetRegions": [
            {
              "name": "[variables('location')]",
              "regionalReplicaCount": "1"
            }
          ]
        },
        "storageProfile": {
          "osDiskImage": {
            "source": {
              "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
              "uri": "[parameters('vhdBlobURL')]"
            }
          }
        }
      }
    }
  ]
}
}
}

```



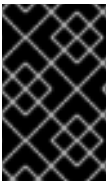
7.11.14. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

7.11.14.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 7.38. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 7.39. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 7.40. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

7.11.15. Creating networking and load balancing components in Azure

You must configure networking and load balancing in Microsoft Azure for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.



NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.

Procedure

1. Copy the template from the **ARM template for the network and load balancers** section of this topic and save it as **03_infra.json** in your cluster's installation directory. This template describes the networking and load balancing objects that your cluster requires.
2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1 The name of the private DNS zone.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Create an **api** DNS record in the public zone for the API public load balancer. The **\${BASE_DOMAIN_RESOURCE_GROUP}** variable must point to the resource group where the public DNS zone exists.

- a. Export the following variable:

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

- b. Create the **api** DNS record in a new public zone:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

If you are adding the cluster to an existing public zone, you can create the **api** DNS record in it instead:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

7.11.15.1. ARM template for the network and load balancers

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

Example 7.53. 03_infra.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  }
}
```

```

},
"vnetBaseName": {
  "type": "string",
  "defaultValue": "",
  "metadata": {
    "description": "The specific customer vnet's base name (optional)"
  }
},
"privateDNSZoneName": {
  "type": "string",
  "metadata": {
    "description": "Name of the private DNS zone"
  }
}
},
"variables": {
  "location": "[resourceGroup().location]",
  "virtualNetworkName": "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
  "virtualNetworkID": "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
  "masterSubnetName": "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
  "masterSubnetRef": "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
  "masterPublicIpAddressName": "[concat(parameters('baseName'), '-master-pip')]",
  "masterPublicIpAddressID": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
  "masterLoadBalancerName": "[concat(parameters('baseName'), '-public-lb')]",
  "masterLoadBalancerID": "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
  "internalLoadBalancerName": "[concat(parameters('baseName'), '-internal-lb')]",
  "internalLoadBalancerID": "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
  "skuName": "Standard"
},
"resources": [
{
  "apiVersion": "2018-12-01",
  "type": "Microsoft.Network/publicIPAddresses",
  "name": "[variables('masterPublicIpAddressName')]",
  "location": "[variables('location')]",
  "sku": {
    "name": "[variables('skuName')]"
  },
  "properties": {
    "publicIPAllocationMethod": "Static",
    "dnsSettings": {
      "domainNameLabel": "[variables('masterPublicIpAddressName')]"
    }
  }
},
{
  "apiVersion": "2018-12-01",
  "type": "Microsoft.Network/loadBalancers",
  "name": "[variables('masterLoadBalancerName')]",

```

```

"location" : "[variables('location')]",
"sku": {
  "name": "[variables('skuName')]"
},
"dependsOn" : [
  "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
],
"properties" : {
  "frontendIPConfigurations" : [
    {
      "name" : "public-lb-ip",
      "properties" : {
        "publicIPAddress" : {
          "id" : "[variables('masterPublicIpAddressID')]"
        }
      }
    }
  ],
  "backendAddressPools" : [
    {
      "name" : "public-lb-backend"
    }
  ],
  "loadBalancingRules" : [
    {
      "name" : "api-internal",
      "properties" : {
        "frontendIPConfiguration" : {
          "id" : "[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-
ip')]"
        },
        "backendAddressPool" : {
          "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/public-lb-
backend')]"
        },
        "protocol" : "Tcp",
        "loadDistribution" : "Default",
        "idleTimeoutInMinutes" : 30,
        "frontendPort" : 6443,
        "backendPort" : 6443,
        "probe" : {
          "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
        }
      }
    }
  ],
  "probes" : [
    {
      "name" : "api-internal-probe",
      "properties" : {
        "protocol" : "Https",
        "port" : 6443,
        "requestPath" : "/readyz",
        "intervalInSeconds" : 10,
        "numberOfProbes" : 3
      }
    }
  ]
}

```

```

    }
  ]
}
},
{
  "apiVersion": "2018-12-01",
  "type": "Microsoft.Network/loadBalancers",
  "name": "[variables('internalLoadBalancerName')]",
  "location": "[variables('location')]",
  "sku": {
    "name": "[variables('skuName')]"
  },
  "properties": {
    "frontendIPConfigurations": [
      {
        "name": "internal-lb-ip",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "subnet": {
            "id": "[variables('masterSubnetRef')]"
          },
          "privateIPAddressVersion": "IPv4"
        }
      }
    ],
    "backendAddressPools": [
      {
        "name": "internal-lb-backend"
      }
    ],
    "loadBalancingRules": [
      {
        "name": "api-internal",
        "properties": {
          "frontendIPConfiguration": {
            "id": "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip)"]"
          },
          "frontendPort": 6443,
          "backendPort": 6443,
          "enableFloatingIP": false,
          "idleTimeoutInMinutes": 30,
          "protocol": "Tcp",
          "enableTcpReset": false,
          "loadDistribution": "Default",
          "backendAddressPool": {
            "id": "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend)"]"
          },
          "probe": {
            "id": "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe)"]"
          }
        }
      }
    ],
    {
      "name": "sint",

```

```

    "properties" : {
      "frontendIPConfiguration" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
      },
      "frontendPort" : 22623,
      "backendPort" : 22623,
      "enableFloatingIP" : false,
      "idleTimeoutInMinutes" : 30,
      "protocol" : "Tcp",
      "enableTcpReset" : false,
      "loadDistribution" : "Default",
      "backendAddressPool" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
      },
      "probe" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
      }
    }
  ],
  "probes" : [
    {
      "name" : "api-internal-probe",
      "properties" : {
        "protocol" : "Https",
        "port" : 6443,
        "requestPath" : "/readyz",
        "intervalInSeconds" : 10,
        "numberOfProbes" : 3
      }
    },
    {
      "name" : "sint-probe",
      "properties" : {
        "protocol" : "Https",
        "port" : 22623,
        "requestPath" : "/healthz",
        "intervalInSeconds" : 10,
        "numberOfProbes" : 3
      }
    }
  ]
},
{
  "apiVersion": "2018-09-01",
  "type": "Microsoft.Network/privateDnsZones/A",
  "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
  "location": "[variables('location')]",
  "dependsOn" : [
    "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
  ],
  "properties": {
    "ttl": 60,

```

```

    "aRecords": [
      {
        "ipv4Address": "[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
      }
    ]
  },
  {
    "apiVersion": "2018-09-01",
    "type": "Microsoft.Network/privateDnsZones/A",
    "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
    "location": "[variables('location')]",
    "dependsOn": [
      "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
    ],
    "properties": {
      "ttl": 60,
      "aRecords": [
        {
          "ipv4Address": "[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
        }
      ]
    }
  }
]
}

```

7.11.16. Creating the bootstrap machine in Azure

You must create the bootstrap machine in Microsoft Azure to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.



NOTE

If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.

- Create control plane and compute roles.

Procedure

1. Copy the template from the **ARM template for the bootstrap machine** section of this topic and save it as **04_bootstrap.json** in your cluster's installation directory. This template describes the bootstrap machine that your cluster requires.
2. Export the bootstrap URL variable:

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. Export the bootstrap ignition variable:

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n`
```

4. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1 The bootstrap Ignition content for the bootstrap cluster.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

7.11.16.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 7.54. 04_bootstrap.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
```

```

    "type": "string",
    "defaultValue": "",
    "metadata": {
      "description": "The specific customer vnet's base name (optional)"
    }
  },
  "bootstrapIgnition": {
    "type": "string",
    "minLength": 1,
    "metadata": {
      "description": "Bootstrap ignition content for the bootstrap cluster"
    }
  },
  "sshKeyData": {
    "type": "securestring",
    "defaultValue": "Unused",
    "metadata": {
      "description": "Unused"
    }
  },
  "bootstrapVMSize": {
    "type": "string",
    "defaultValue": "Standard_D4s_v3",
    "metadata": {
      "description": "The size of the Bootstrap Virtual Machine"
    }
  },
  "hyperVGen": {
    "type": "string",
    "metadata": {
      "description": "VM generation image to use"
    },
    "defaultValue": "V2",
    "allowedValues": [
      "V1",
      "V2"
    ]
  },
  "variables": {
    "location": "[resourceGroup().location]",
    "virtualNetworkName": "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID": "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName": "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef": "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName": "[concat(parameters('baseName'), '-public-lb')]",
    "internalLoadBalancerName": "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath": "/home/core/.ssh/authorized_keys",
    "identityName": "[concat(parameters('baseName'), '-identity')]",
    "vmName": "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName": "[concat(variables('vmName'), '-nic')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]"
  }
}

```



```

    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-gen2', ''))]",
    "clusterNsgName" : "[concat(if(not(empty(parameters('vnetBaseName'))), parameters('vnetBaseName'), parameters('baseName')), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('sshPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "Standard"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "name" : "[variables('nicName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
      ],
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "publicIPAddress": {
                "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
              },
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
                },
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
                }
              ]
            }
          }
        ]
      }
    }
  ]
}

```

```

    }
  ]
}
},
{
  "apiVersion": "2018-06-01",
  "type": "Microsoft.Compute/virtualMachines",
  "name": "[variables('vmName')]",
  "location": "[variables('location')]",
  "identity": {
    "type": "userAssigned",
    "userAssignedIdentities": {
      "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]": {}
    }
  },
  "dependsOn": [
    "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
  ],
  "properties": {
    "hardwareProfile": {
      "vmSize": "[parameters('bootstrapVMSize')]"
    },
    "osProfile": {
      "computerName": "[variables('vmName')]",
      "adminUsername": "core",
      "adminPassword": "NotActuallyApplied!",
      "customData": "[parameters('bootstrapIgnition')]",
      "linuxConfiguration": {
        "disablePasswordAuthentication": false
      }
    },
    "storageProfile": {
      "imageReference": {
        "id": "[resourceID('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
      },
      "osDisk": {
        "name": "[concat(variables('vmName'),'_OSDisk')]",
        "osType": "Linux",
        "createOption": "FromImage",
        "managedDisk": {
          "storageAccountType": "Premium_LRS"
        },
        "diskSizeGB": 100
      }
    },
    "networkProfile": {
      "networkInterfaces": [
        {
          "id": "[resourceID('Microsoft.Network/networkInterfaces', variables('nicName'))]"
        }
      ]
    }
  }
},
},

```

```

{
  "apiVersion" : "2018-06-01",
  "type": "Microsoft.Network/networkSecurityGroups/securityRules",
  "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
  "location" : "[variables('location')]",
  "dependsOn" : [
    "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
  ],
  "properties": {
    "protocol" : "Tcp",
    "sourcePortRange" : "*",
    "destinationPortRange" : "22",
    "sourceAddressPrefix" : "*",
    "destinationAddressPrefix" : "*",
    "access" : "Allow",
    "priority" : 100,
    "direction" : "Inbound"
  }
}
]
}

```

7.11.17. Creating the control plane machines in Azure

You must create the control plane machines in Microsoft Azure for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.



NOTE

By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's Azure Resource Manager (ARM) template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the **ARM template for control plane machines** section of this topic and save it as **05_masters.json** in your cluster's installation directory. This template describes the control plane machines that your cluster requires.
2. Export the following variable needed by the control plane machine deployment:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1

The Ignition content for the control plane nodes.

2

The base name to be used in resource names; this is usually the cluster's infrastructure ID.

7.11.17.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 7.55. 05_masters.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName" : {
      "type" : "string",
      "defaultValue" : "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "masterIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the master nodes"
      }
    },
    "numberOfMasters" : {
```

```

    "type" : "int",
    "defaultValue" : 3,
    "minValue" : 2,
    "maxValue" : 30,
    "metadata" : {
      "description" : "Number of OpenShift masters to deploy"
    }
  },
  "sshKeyData" : {
    "type" : "securestring",
    "defaultValue" : "Unused",
    "metadata" : {
      "description" : "Unused"
    }
  },
  "privateDNSZoneName" : {
    "type" : "string",
    "defaultValue" : "",
    "metadata" : {
      "description" : "unused"
    }
  },
  "masterVMSize" : {
    "type" : "string",
    "defaultValue" : "Standard_D8s_v3",
    "metadata" : {
      "description" : "The size of the Master Virtual Machines"
    }
  },
  "diskSizeGB" : {
    "type" : "int",
    "defaultValue" : 1024,
    "metadata" : {
      "description" : "Size of the Master VM OS disk, in GB"
    }
  },
  "hyperVGen" : {
    "type" : "string",
    "metadata" : {
      "description" : "VM generation image to use"
    },
    "defaultValue" : "V2",
    "allowedValues" : [
      "V1",
      "V2"
    ]
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",

```

```

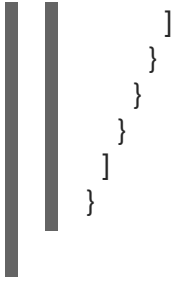
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'), '-public-lb')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" : "[parameters('numberOfMasters')]",
        "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
      "location" : "[variables('location')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
                },
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
                }
              ]
            }
          }
        ]
      }
    }
  ],
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",

```

```

"copy" : {
  "name" : "vmCopy",
  "count" : "[length(variables('vmNames'))]"
},
"name" : "[variables('vmNames')[copyIndex()]]",
"location" : "[variables('location')]",
"identity" : {
  "type" : "userAssigned",
  "userAssignedIdentities" : {
    "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
  }
},
"dependsOn" : [
  "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
],
"properties" : {
  "hardwareProfile" : {
    "vmSize" : "[parameters('masterVMSize')]"
  },
  "osProfile" : {
    "computerName" : "[variables('vmNames')[copyIndex()]]",
    "adminUsername" : "core",
    "adminPassword" : "NotActuallyApplied!",
    "customData" : "[parameters('masterIgnition')]",
    "linuxConfiguration" : {
      "disablePasswordAuthentication" : false
    }
  },
  "storageProfile" : {
    "imageReference": {
      "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
    },
    "osDisk" : {
      "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
      "osType" : "Linux",
      "createOption" : "FromImage",
      "caching": "ReadOnly",
      "writeAcceleratorEnabled": false,
      "managedDisk": {
        "storageAccountType": "Premium_LRS"
      },
      "diskSizeGB" : "[parameters('diskSizeGB')]"
    }
  },
  "networkProfile" : {
    "networkInterfaces" : [
      {
        "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
        "properties": {
          "primary": false
        }
      }
    ]
  }
}

```



7.11.18. Wait for bootstrap completion and remove bootstrap resources in Azure

After you create all of the required infrastructure in Microsoft Azure, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
```



```
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip
```



NOTE

If you do not delete the bootstrap server, installation may not succeed due to API traffic being routed to the bootstrap server.

7.11.19. Creating additional worker machines in Azure

You can create worker machines in Microsoft Azure for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type **06_workers.json** in the file.



NOTE

By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's ARM template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster's installation directory. This template describes the worker machines that your cluster requires.

- Export the following variable needed by the worker machine deployment:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d "\n`
```

- Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \  
  --template-file "<installation_directory>/06_workers.json" \  
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1 \  
  --parameters baseName="${INFRA_ID}" 2
```

- The Ignition content for the worker nodes.
- The base name to be used in resource names; this is usually the cluster's infrastructure ID.

7.11.19.1. ARM template for worker machines

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 7.56. 06_workers.json ARM template

```
{  
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
  "contentVersion" : "1.0.0.0",  
  "parameters" : {  
    "baseName" : {  
      "type" : "string",  
      "minLength" : 1,  
      "metadata" : {  
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"  
      }  
    },  
    "vnetBaseName" : {  
      "type" : "string",  
      "defaultValue" : "",  
      "metadata" : {  
        "description" : "The specific customer vnet's base name (optional)"  
      }  
    },  
    "workerIgnition" : {  
      "type" : "string",  
      "metadata" : {  
        "description" : "Ignition content for the worker nodes"  
      }  
    },  
    "numberOfNodes" : {  
      "type" : "int",  
      "defaultValue" : 3,  
      "minValue" : 2,  
      "maxValue" : 30,  
      "metadata" : {  
        "description" : "Number of OpenShift compute nodes to deploy"  
      }  
    }  
  }  
}
```

```

    }
  },
  "sshKeyData" : {
    "type" : "securestring",
    "defaultValue" : "Unused",
    "metadata" : {
      "description" : "Unused"
    }
  },
  "nodeVMSize" : {
    "type" : "string",
    "defaultValue" : "Standard_D4s_v3",
    "metadata" : {
      "description" : "The size of the each Node Virtual Machine"
    }
  },
  "hyperVGen": {
    "type": "string",
    "metadata": {
      "description": "VM generation image to use"
    },
    "defaultValue": "V2",
    "allowedValues": [
      "V1",
      "V2"
    ]
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" : "[parameters('numberOfNodes')]",
        "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2019-05-01",

```

```

"name" : "[concat('node', copyIndex())]",
"type" : "Microsoft.Resources/deployments",
"copy" : {
  "name" : "nodeCopy",
  "count" : "[length(variables('vmNames'))]"
},
"properties" : {
  "mode" : "Incremental",
  "template" : {
    "$schema" : "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion" : "1.0.0.0",
    "resources" : [
      {
        "apiVersion" : "2018-06-01",
        "type" : "Microsoft.Network/networkInterfaces",
        "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
        "location" : "[variables('location')]",
        "properties" : {
          "ipConfigurations" : [
            {
              "name" : "pipConfig",
              "properties" : {
                "privateIPAllocationMethod" : "Dynamic",
                "subnet" : {
                  "id" : "[variables('nodeSubnetRef')]"
                }
              }
            }
          ]
        }
      }
    ]
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "name" : "[variables('vmNames')[copyIndex()]]",
    "location" : "[variables('location')]",
    "tags" : {
      "kubernetes.io-cluster-ffranzupi": "owned"
    },
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('nodeVMSize')]"
      },
      "osProfile" : {

```

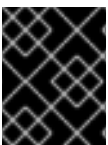
```

"computerName" : "[variables('vmNames')[copyIndex()]]",
"adminUsername" : "capi",
"adminPassword" : "NotActuallyApplied!",
"customData" : "[parameters('workerIgnition')]",
"linuxConfiguration" : {
  "disablePasswordAuthentication" : false
}
},
"storageProfile" : {
  "imageReference": {
    "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
  },
  "osDisk" : {
    "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
    "osType" : "Linux",
    "createOption" : "FromImage",
    "managedDisk": {
      "storageAccountType": "Premium_LRS"
    },
    "diskSizeGB": 128
  }
},
"networkProfile" : {
  "networkInterfaces" : [
    {
      "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
      "properties": {
        "primary": true
      }
    }
  ]
}
]
}
}
]
}
}
]
}
}

```

7.11.20. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

7.11.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

7.11.22. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```


In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```

NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...

```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```

NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   73m   v1.25.0
master-1  Ready   master   73m   v1.25.0
master-2  Ready   master   74m   v1.25.0
worker-0  Ready   worker   11m   v1.25.0
worker-1  Ready   worker   11m   v1.25.0

```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

7.11.23. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Microsoft Azure by using infrastructure that you provisioned.
- Install the OpenShift CLI (**oc**).
- Install or update the [Azure CLI](#).

Procedure

1. Confirm the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
router-default LoadBalancer  172.30.20.10  35.130.120.110 80:32288/TCP,443:31215/TCP 20
```

2. Export the Ingress router IP as a variable:

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

3. Add a ***.apps** record to the public DNS zone.

- a. If you are adding this cluster to a new public zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

- b. If you are adding this cluster to an already existing public zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

4. Add a ***.apps** record to the private DNS zone:

- a. Create a ***.apps** record by using the following command:

```
$ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
```

- b. Add the ***.apps** record to the private DNS zone by using the following command:

```
$ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

7.11.24. Completing an Azure installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure infrastructure.
- Install the **oc** CLI and log in.

Procedure

- Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

7.11.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use [subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

7.12. UNINSTALLING A CLUSTER ON AZURE

You can remove a cluster that you deployed to Microsoft Azure.

7.12.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

While you can uninstall the cluster using the copy of the installation program that was used to deploy it, using OpenShift Container Platform version 4.13 or later is recommended.

The removal of service principals is dependent on the Microsoft Azure AD Graph API. Using version 4.13 or later of the installation program ensures that service principals are removed without the need for manual intervention, if and when Microsoft decides to [retire](#) the Azure AD Graph API.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```

$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2

```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 8. INSTALLING ON AZURE STACK HUB

8.1. PREPARING TO INSTALL ON AZURE STACK HUB

8.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You have installed Azure Stack Hub version 2008 or later.

8.1.2. Requirements for installing OpenShift Container Platform on Azure Stack Hub

Before installing OpenShift Container Platform on Microsoft Azure Stack Hub, you must configure an Azure account.

See [Configuring an Azure Stack Hub account](#) for details about account configuration, account limits, DNS zone configuration, required roles, and creating service principals.

8.1.3. Choosing a method to install OpenShift Container Platform on Azure Stack Hub

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

8.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on Azure Stack Hub infrastructure that is provisioned by the OpenShift Container Platform installation program, by using the following method:

- [Installing a cluster on Azure Stack Hub with an installer-provisioned infrastructure](#) You can install OpenShift Container Platform on Azure Stack Hub infrastructure that is provisioned by the OpenShift Container Platform installation program.

8.1.3.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on Azure Stack Hub infrastructure that you provision, by using the following method:

- [Installing a cluster on Azure Stack Hub using ARM templates](#) You can install OpenShift Container Platform on Azure Stack Hub by using infrastructure that you provide. You can use the provided Azure Resource Manager (ARM) templates to assist with an installation.

8.1.4. Next steps

- [Configuring an Azure Stack Hub account](#)

8.2. CONFIGURING AN AZURE STACK HUB ACCOUNT

Before you can install OpenShift Container Platform, you must configure a Microsoft Azure account.



IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

8.2.1. Azure Stack Hub account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure Stack Hub components, and the default [Quota types in Azure Stack Hub](#) affect your ability to install OpenShift Container Platform clusters.

The following table summarizes the Azure Stack Hub components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Description
vCPU	56	<p>A default cluster requires 56 vCPUs, so you must increase the account limit.</p> <p>By default, each cluster creates the following instances:</p> <ul style="list-style-type: none"> • One bootstrap machine, which is removed after installation • Three control plane machines • Three compute machines <p>Because the bootstrap, control plane, and worker machines use Standard_DS4_v2 virtual machines, which use 8 vCPUs, a default cluster requires 56 vCPUs. The bootstrap node VM is used only during installation.</p> <p>To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.</p>
VNet	1	Each default cluster requires one Virtual Network (VNet), which contains two subnets.
Network interfaces	7	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.

Component	Number of components required by default	Description						
Network security groups	2	<p>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</p> <table border="1"> <tr> <td>control plane</td> <td>Allows the control plane machines to be reached on port 6443 from anywhere</td> </tr> <tr> <td>node</td> <td>Allows worker nodes to be reached from the internet on ports 80 and 443</td> </tr> </table>	control plane	Allows the control plane machines to be reached on port 6443 from anywhere	node	Allows worker nodes to be reached from the internet on ports 80 and 443		
control plane	Allows the control plane machines to be reached on port 6443 from anywhere							
node	Allows worker nodes to be reached from the internet on ports 80 and 443							
Network load balancers	3	<p>Each cluster creates the following load balancers:</p> <table border="1"> <tr> <td>default</td> <td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td> </tr> <tr> <td>internal</td> <td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td> </tr> <tr> <td>external</td> <td>Public IP address that load balances requests to port 6443 across control plane machines</td> </tr> </table> <p>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</p>	default	Public IP address that load balances requests to ports 80 and 443 across worker machines	internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines	external	Public IP address that load balances requests to port 6443 across control plane machines
default	Public IP address that load balances requests to ports 80 and 443 across worker machines							
internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines							
external	Public IP address that load balances requests to port 6443 across control plane machines							
Public IP addresses	2	The public load balancer uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.						
Private IP addresses	7	The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.						

Additional resources

- [Optimizing storage](#).

8.2.2. Configuring a DNS zone in Azure Stack Hub

To successfully install OpenShift Container Platform on Azure Stack Hub, you must create DNS records in an Azure Stack Hub DNS zone. The DNS zone must be authoritative for the domain. To delegate a registrar's DNS zone to Azure Stack Hub, see Microsoft's documentation for [Azure Stack Hub datacenter DNS integration](#).

8.2.3. Required Azure Stack Hub roles

Your Microsoft Azure Stack Hub account must have the following roles for the subscription that you use:

- **Owner**

To set roles on the Azure portal, see the [Manage access to resources in Azure Stack Hub with role-based access control](#) in the Microsoft documentation.

8.2.4. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the [Azure CLI](#).
- Your Azure account has the required roles for the subscription that you use.

Procedure

1. Register your environment:

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

- 1 Specify the Azure Resource Manager endpoint, ``https://management.<region>.<fqdn>/``.

See the [Microsoft documentation](#) for details.

2. Set the active environment:

```
$ az cloud set -n AzureStackCloud
```

3. Update your environment configuration to use the specific API version for Azure Stack Hub:

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. Log in to the Azure CLI:

```
$ az login
```

If you are in a multitenant environment, you must also supply the tenant ID.

5. If your Azure account uses subscriptions, ensure that you are using the right subscription:

- a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": AzureStackCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

- b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", 1
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

- 1 Ensure that the value of the **tenantId** parameter is the correct subscription ID.

- c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> 1
```

- 1 Specify the subscription ID.

- d. Verify the subscription ID update:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
```

```

    "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
}

```

6. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.
7. Create the service principal for your account:

```

$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ 1
--scopes /subscriptions/<subscription_id> 2
--years <years> 3

```

- 1 Specify the service principal name.
- 2 Specify the subscription ID.
- 3 Specify the number of years. By default, a service principal expires in one year. By using the **--years** option you can extend the validity of your service principal.

Example output

```

Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>,
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}

```

8. Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

Additional resources

- For more information about CCO modes, see [About the Cloud Credential Operator](#).

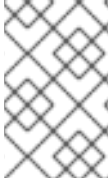
8.2.5. Next steps

- Install an OpenShift Container Platform cluster:
 - [Installing a cluster quickly on Azure Stack Hub](#) .

- Install an OpenShift Container Platform cluster on Azure Stack Hub with user-provisioned infrastructure by following [Installing a cluster on Azure Stack Hub using ARM templates](#) .

8.3. INSTALLING A CLUSTER ON AZURE STACK HUB WITH AN INSTALLER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on Microsoft Azure Stack Hub with an installer-provisioned infrastructure. However, you must manually configure the **install-config.yaml** file to specify values that are specific to Azure Stack Hub.



NOTE

While you can select **azure** when using the installation program to deploy a cluster using installer-provisioned infrastructure, this option is only supported for the Azure Public Cloud.

8.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure Stack Hub account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You verified that you have approximately 16 GB of local disk space. Installing the cluster requires that you download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment. Decompressing the VHD files requires this amount of local disk space.

8.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

8.3.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

8.3.4. Uploading the RHCOS cluster image

You must download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment.

Prerequisites

- Configure an Azure account.

Procedure

1. Obtain the RHCOS VHD cluster image:

- a. Export the URL of the RHCOS VHD to an environment variable.

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

- b. Download the compressed RHCOS VHD file locally.

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. Decompress the VHD file.



NOTE

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. The VHD file can be deleted once you have uploaded it.

3. Upload the local VHD to the Azure Stack Hub environment, making sure that the blob is publicly available. For example, you can upload the VHD to a blob using the **az** cli or the web portal.

8.3.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

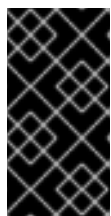
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select **Azure** as the cloud provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

8.3.6. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

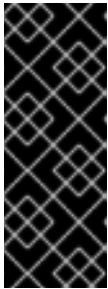
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



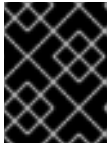
NOTE

You must name this configuration file **install-config.yaml**.

Make the following modifications:

- a. Specify the required installation parameters.
- b. Update the **platform.azure** section to specify the parameters that are specific to Azure Stack Hub.

- c. Optional: Update one or more of the default configuration parameters to customize the installation.
For more information about the parameters, see "Installation configuration parameters".
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

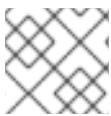


IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

8.3.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

8.3.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 8.1. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

8.3.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 8.2. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

8.3.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:



Table 8.3. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}

Parameter	Description	Values
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div data-bbox="485 1386 592 1704" data-label="Image"> </div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 20px; border: 1px solid black; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 20px; border: 1px solid black; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="display: flex; align-items: center;">  <div style="margin-left: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div data-bbox="488 1442 592 1637" style="display: flex; align-items: center; margin-top: 10px;">  <div style="margin-left: 10px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

8.3.6.1.4. Additional Azure Stack Hub configuration parameters

Additional Azure configuration parameters are described in the following table:

Table 8.4. Additional Azure Stack Hub parameters

Parameter	Description	Values
compute.platform.azure.osDisk.diskSize GB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS or premium_LRS . The default is premium_LRS .

Parameter	Description	Values
compute.platform.azure.type	Defines the azure instance type for compute machines.	String
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS .
controlPlane.platform.azure.type	Defines the azure instance type for control plane machines.	String
platform.azure.defaultMachinePlatform.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.osDisk.diskType	Defines the type of disk.	standard_LRS or premium_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.armEndpoint	The URL of the Azure Resource Manager endpoint that your Azure Stack Hub operator provides.	String
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .
platform.azure.region	The name of your Azure Stack Hub local region.	String

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints.	AzureStackCloud
clusterOSImage	The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.	String, for example, <code>https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-azurestack.x86_64.vhd</code>

8.3.6.2. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

apiVersion: v1

baseDomain: example.com **1**

```

credentialsMode: Manual
controlPlane: 2 3
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 4
        diskType: premium_LRS
    replicas: 3
compute: 5
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 6
        diskType: premium_LRS
    replicas: 3
metadata:
  name: test-cluster 7 8
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint 10 11
    baseDomainResourceGroupName: resource_group 12 13
    region: azure_stack_local_region 14 15
    resourceGroupName: existing_resource_group 16
    outboundType: Loadbalancer
    cloudName: AzureStackCloud 17
    clusterOSImage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
    azurestack.x86_64.vhd 18 19
  pullSecret: '{"auths": ...}' 20 21
  fips: false 22
  sshKey: ssh-ed25519 AAAA... 23
  additionalTrustBundle: | 24
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----

```

1 7 10 12 14 17 18 20 Required.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

3 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both

sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

- 4 6 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 8 The name of the cluster.
- 9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 11 The Azure Resource Manager endpoint that your Azure Stack Hub operator provides.
- 13 The name of the resource group that contains the DNS zone for your base domain.
- 15 The name of your Azure Stack Hub local region.
- 16 The name of an existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 19 The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.
- 21 The pull secret required to authenticate your cluster.
- 22 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS validated or Modules In Progress cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 23 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 24 If the Azure Stack Hub environment is using an internal Certificate Authority (CA), adding the CA certificate is required.

8.3.7. Manually manage cloud credentials

The Cloud Credential Operator (CCO) only supports your cloud provider in manual mode. As a result, you must specify the identity and access management (IAM) secrets for your cloud provider.

Procedure

1. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

2. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use by running the following command:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

3. Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 \
  --credentials-requests \
  --cloud=azure
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
    ...
```

4. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
```

```
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
      ...
  secretRef:
    name: <component-secret>
    namespace: <component-namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component-secret>
  namespace: <component-namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

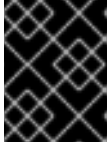
- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- To find **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation, run the following command:

```
$ grep "release.openshift.io/feature-set" *
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```


**IMPORTANT**

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- [Updating cloud provider resources with manually maintained credentials](#)

8.3.8. Configuring the cluster to use an internal CA

If the Azure Stack Hub environment is using an internal Certificate Authority (CA), update the **cluster-proxy-01-config.yaml** file to configure the cluster to use the internal CA.

Prerequisites

- Create the **install-config.yaml** file and specify the certificate trust bundle in **.pem** format.
- Create the cluster manifests.

Procedure

1. From the directory in which the installation program creates files, go to the **manifests** directory.
2. Add **user-ca-bundle** to the **spec.trustedCA.name** field.

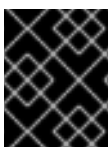
Example cluster-proxy-01-config.yaml file

```
apiVersion: config.openshift.io/v1
kind: Proxy
metadata:
  creationTimestamp: null
  name: cluster
spec:
  trustedCA:
    name: user-ca-bundle
status: {}
```

3. Optional: Back up the **manifests/ cluster-proxy-01-config.yaml** file. The installation program consumes the **manifests/** directory when you deploy the cluster.

8.3.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.

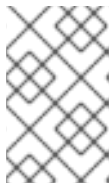
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



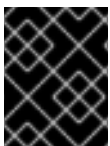
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

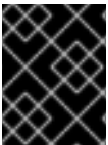


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

8.3.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

8.3.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

8.3.12. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

- List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console    console-openshift-console.apps.<cluster_name>.<base_domain>    console
https reencrypt/Redirect None
```

- Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- [Accessing the web console](#)

8.3.13. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

8.3.14. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

8.4. INSTALLING A CLUSTER ON AZURE STACK HUB WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Azure Stack Hub. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.



NOTE

While you can select **azure** when using the installation program to deploy a cluster using installer-provisioned infrastructure, this option is only supported for the Azure Public Cloud.

8.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure Stack Hub account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You verified that you have approximately 16 GB of local disk space. Installing the cluster requires that you download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment. Decompressing the VHD files requires this amount of local disk space.

8.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

8.4.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

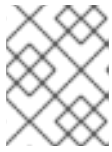
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```


3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

**NOTE**

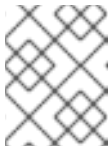
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

8.4.4. Uploading the RHCOS cluster image

You must download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment.

Prerequisites

- Configure an Azure account.

Procedure

1. Obtain the RHCOS VHD cluster image:

- a. Export the URL of the RHCOS VHD to an environment variable.

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

- b. Download the compressed RHCOS VHD file locally.

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. Decompress the VHD file.



NOTE

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. The VHD file can be deleted once you have uploaded it.

3. Upload the local VHD to the Azure Stack Hub environment, making sure that the blob is publicly available. For example, you can upload the VHD to a blob using the **az** cli or the web portal.

8.4.5. Obtaining the installation program

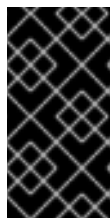
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select **Azure** as the cloud provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

8.4.6. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



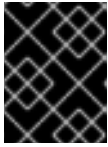
NOTE

You must name this configuration file **install-config.yaml**.

Make the following modifications:

- a. Specify the required installation parameters.
- b. Update the **platform.azure** section to specify the parameters that are specific to Azure Stack Hub.

- c. Optional: Update one or more of the default configuration parameters to customize the installation.
For more information about the parameters, see "Installation configuration parameters".
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

8.4.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

8.4.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 8.5. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

8.4.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 8.6. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

8.4.6.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 8.7. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}

Parameter	Description	Values
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div data-bbox="485 1386 592 1704" data-label="Image"> </div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code>
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string (<code>""</code>).

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 150px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 60px; height: 60px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

8.4.6.1.4. Additional Azure Stack Hub configuration parameters

Additional Azure configuration parameters are described in the following table:

Table 8.8. Additional Azure Stack Hub parameters

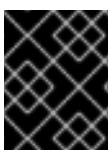
Parameter	Description	Values
compute.platform.azure.osDisk.diskSize GB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
compute.platform.azure.osDisk.diskType	Defines the type of disk.	standard_LRS or premium_LRS . The default is premium_LRS .

Parameter	Description	Values
compute.platform.azure.type	Defines the azure instance type for compute machines.	String
controlPlane.platform.azure.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 1024 .
controlPlane.platform.azure.osDisk.diskType	Defines the type of disk.	premium_LRS .
controlPlane.platform.azure.type	Defines the azure instance type for control plane machines.	String
platform.azure.defaultMachinePlatform.osDisk.diskSizeGB	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is 128 .
platform.azure.defaultMachinePlatform.osDisk.diskType	Defines the type of disk.	standard_LRS or premium_LRS . The default is premium_LRS .
platform.azure.defaultMachinePlatform.type	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform.azure.armEndpoint	The URL of the Azure Resource Manager endpoint that your Azure Stack Hub operator provides.	String
platform.azure.baseDomainResourceGroupName	The name of the resource group that contains the DNS zone for your base domain.	String, for example production_cluster .
platform.azure.region	The name of your Azure Stack Hub local region.	String

Parameter	Description	Values
platform.azure.resourceGroupName	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example existing_resource_group .
platform.azure.outboundType	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.	LoadBalancer or UserDefinedRouting . The default is LoadBalancer .
platform.azure.cloudName	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints.	AzureStackCloud
clusterOSImage	The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.	String, for example, <code>https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-azurestack.x86_64.vhd</code>

8.4.6.2. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

apiVersion: v1

baseDomain: example.com **1**

```

credentialsMode: Manual
controlPlane: 2 3
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 4
        diskType: premium_LRS
      replicas: 3
compute: 5
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 6
        diskType: premium_LRS
      replicas: 3
metadata:
  name: test-cluster 7 8
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint 10 11
    baseDomainResourceGroupName: resource_group 12 13
    region: azure_stack_local_region 14 15
    resourceGroupName: existing_resource_group 16
    outboundType: Loadbalancer
    cloudName: AzureStackCloud 17
    clusterOSImage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
azurestack.x86_64.vhd 18 19
pullSecret: '{"auths": ...}' 20 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
additionalTrustBundle: | 24
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----

```

1 7 10 12 14 17 18 20 Required.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

3 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both

sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

- 4 6 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 8 The name of the cluster.
- 9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 11 The Azure Resource Manager endpoint that your Azure Stack Hub operator provides.
- 13 The name of the resource group that contains the DNS zone for your base domain.
- 15 The name of your Azure Stack Hub local region.
- 16 The name of an existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 19 The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.
- 21 The pull secret required to authenticate your cluster.
- 22 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS validated or Modules In Progress cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 23 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 24 If the Azure Stack Hub environment is using an internal Certificate Authority (CA), adding the CA certificate is required.

8.4.7. Manually manage cloud credentials

The Cloud Credential Operator (CCO) only supports your cloud provider in manual mode. As a result, you must specify the identity and access management (IAM) secrets for your cloud provider.

Procedure

1. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

2. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use by running the following command:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

3. Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 \
  --credentials-requests \
  --cloud=azure
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
    ...
```

4. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
```

```
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
      ...
  secretRef:
    name: <component-secret>
    namespace: <component-namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component-secret>
  namespace: <component-namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```



IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

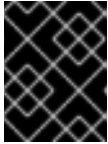
- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- To find **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation, run the following command:

```
$ grep "release.openshift.io/feature-set" *
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```



IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- [Updating a cluster using the web console](#)
- [Updating a cluster using the CLI](#)

8.4.8. Configuring the cluster to use an internal CA

If the Azure Stack Hub environment is using an internal Certificate Authority (CA), update the **cluster-proxy-01-config.yaml** file to configure the cluster to use the internal CA.

Prerequisites

- Create the **install-config.yaml** file and specify the certificate trust bundle in **.pem** format.
- Create the cluster manifests.

Procedure

1. From the directory in which the installation program creates files, go to the **manifests** directory.
2. Add **user-ca-bundle** to the **spec.trustedCA.name** field.

Example cluster-proxy-01-config.yaml file

```
apiVersion: config.openshift.io/v1
kind: Proxy
metadata:
  creationTimestamp: null
  name: cluster
spec:
  trustedCA:
    name: user-ca-bundle
status: {}
```

3. Optional: Back up the **manifests/ cluster-proxy-01-config.yaml** file. The installation program consumes the **manifests/** directory when you deploy the cluster.

8.4.9. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**

- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

8.4.10. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the `<installation_directory>/manifests/` directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

8.4.11. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

8.4.11.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:


Table 8.9. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 8.10. defaultNetwork object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 8.11. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>

Field	Type	Description
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 8.12. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.

Field	Type	Description
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.  NOTE While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 8.13. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 8.14. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

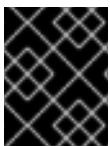
The values for the **kubeProxyConfig** object are defined in the following table:

Table 8.15. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

8.4.12. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.



IMPORTANT

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

where:

<installation_directory>

Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  EOF
```

where:

<installation_directory>

Specifies the directory name that contains the **manifests/** directory for your cluster.

3. Open the **cluster-network-03-config.yml** file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

Specify a hybrid networking configuration

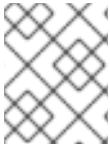
```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
        hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

- 1 Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR cannot overlap with the **clusterNetwork** CIDR.
- 2 Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **4789** port. For more information on this requirement, see the Microsoft documentation on [Pod-to-pod connectivity between hosts is broken](#).

**NOTE**

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.
5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.

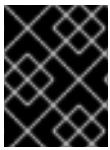


NOTE

For more information on using Linux and Windows nodes in the same cluster, see [Understanding Windows container workloads](#).

8.4.13. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

8.4.14. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

8.4.15. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

■

```
system:admin
```

8.4.16. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

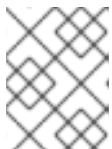


NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/openshift_install.log** log file on the installation host.

Example output

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- [Accessing the web console.](#)

8.4.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#)

[Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

8.4.18. Next steps

- [Validating an installation](#).
- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, you can [remove cloud provider credentials](#).

8.5. INSTALLING A CLUSTER ON AZURE STACK HUB USING ARM TEMPLATES

In OpenShift Container Platform version 4.12, you can install a cluster on Microsoft Azure Stack Hub by using infrastructure that you provide.

Several [Azure Resource Manager](#) (ARM) templates are provided to assist in completing these steps or to help model your own.

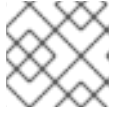


IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several ARM templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

8.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an Azure Stack Hub account](#) to host the cluster.
- You downloaded the Azure CLI and installed it on your computer. See [Install the Azure CLI](#) in the Azure documentation. The documentation below was tested using version **2.28.0** of the Azure CLI. Azure CLI commands might perform differently based on the version you use.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

8.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

8.5.3. Configuring your Azure Stack Hub project

Before you can install OpenShift Container Platform, you must configure an Azure project to host it.

**IMPORTANT**

All Azure Stack Hub resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure Stack Hub restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

8.5.3.1. Azure Stack Hub account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure Stack Hub components, and the default [Quota types in Azure Stack Hub](#) affect your ability to install OpenShift Container Platform clusters.

The following table summarizes the Azure Stack Hub components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Description

Component	Number of components required by default	Description				
vCPU	56	<p>A default cluster requires 56 vCPUs, so you must increase the account limit.</p> <p>By default, each cluster creates the following instances:</p> <ul style="list-style-type: none"> ● One bootstrap machine, which is removed after installation ● Three control plane machines ● Three compute machines <p>Because the bootstrap, control plane, and worker machines use Standard_DS4_v2 virtual machines, which use 8 vCPUs, a default cluster requires 56 vCPUs. The bootstrap node VM is used only during installation.</p> <p>To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.</p>				
VNet	1	Each default cluster requires one Virtual Network (VNet), which contains two subnets.				
Network interfaces	7	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.				
Network security groups	2	<p>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</p> <table border="1"> <tbody> <tr> <td>control plane</td> <td>Allows the control plane machines to be reached on port 6443 from anywhere</td> </tr> <tr> <td>node</td> <td>Allows worker nodes to be reached from the internet on ports 80 and 443</td> </tr> </tbody> </table>	control plane	Allows the control plane machines to be reached on port 6443 from anywhere	node	Allows worker nodes to be reached from the internet on ports 80 and 443
control plane	Allows the control plane machines to be reached on port 6443 from anywhere					
node	Allows worker nodes to be reached from the internet on ports 80 and 443					

Component	Number of components required by default	Description						
Network load balancers	3	<p>Each cluster creates the following load balancers:</p> <table border="1"> <tr> <td>default</td> <td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td> </tr> <tr> <td>internal</td> <td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td> </tr> <tr> <td>external</td> <td>Public IP address that load balances requests to port 6443 across control plane machines</td> </tr> </table> <p>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</p>	default	Public IP address that load balances requests to ports 80 and 443 across worker machines	internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines	external	Public IP address that load balances requests to port 6443 across control plane machines
default	Public IP address that load balances requests to ports 80 and 443 across worker machines							
internal	Private IP address that load balances requests to ports 6443 and 22623 across control plane machines							
external	Public IP address that load balances requests to port 6443 across control plane machines							
Public IP addresses	2	The public load balancer uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.						
Private IP addresses	7	The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.						

Additional resources

- [Optimizing storage](#).

8.5.3.2. Configuring a DNS zone in Azure Stack Hub

To successfully install OpenShift Container Platform on Azure Stack Hub, you must create DNS records in an Azure Stack Hub DNS zone. The DNS zone must be authoritative for the domain. To delegate a registrar's DNS zone to Azure Stack Hub, see Microsoft's documentation for [Azure Stack Hub datacenter DNS integration](#).

You can view Azure's DNS solution by visiting this [example for creating DNS zones](#).

8.5.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

8.5.3.4. Required Azure Stack Hub roles

Your Microsoft Azure Stack Hub account must have the following roles for the subscription that you use:

- **Owner**

To set roles on the Azure portal, see the [Manage access to resources in Azure Stack Hub with role-based access control](#) in the Microsoft documentation.

8.5.3.5. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the [Azure CLI](#).
- Your Azure account has the required roles for the subscription that you use.

Procedure

1. Register your environment:

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

- 1** Specify the Azure Resource Manager endpoint, ``https://management.<region>.<fqdn>/``.

See the [Microsoft documentation](#) for details.

2. Set the active environment:

```
$ az cloud set -n AzureStackCloud
```

3. Update your environment configuration to use the specific API version for Azure Stack Hub:

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. Log in to the Azure CLI:

```
$ az login
```

If you are in a multitenant environment, you must also supply the tenant ID.

5. If your Azure account uses subscriptions, ensure that you are using the right subscription:

- a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

Example output

```
{  
  {
```



```

"cloudName": AzureStackCloud",
"id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
"isDefault": true,
"name": "Subscription Name",
"state": "Enabled",
"tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
"user": {
  "name": "you@example.com",
  "type": "user"
}
}
]

```

- b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```

{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", 1
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}

```

- 1 Ensure that the value of the **tenantId** parameter is the correct subscription ID.

- c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> 1
```

- 1 Specify the subscription ID.

- d. Verify the subscription ID update:

```
$ az account show
```

Example output

```

{
  "environmentName": AzureStackCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",

```

```

    "state": "Enabled",
    "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
}

```

- Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.
- Create the service principal for your account:

```

$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ 1
--scopes /subscriptions/<subscription_id> 2
--years <years> 3

```

- Specify the service principal name.
- Specify the subscription ID.
- Specify the number of years. By default, a service principal expires in one year. By using the **--years** option you can extend the validity of your service principal.

Example output

```

Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>,
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}

```

- Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

Additional resources

- For more information about CCO modes, see [About the Cloud Credential Operator](#).

8.5.4. Obtaining the installation program

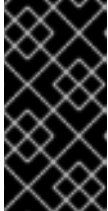
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

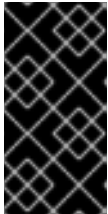
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select **Azure** as the cloud provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

8.5.5. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

8.5.6. Creating the installation files for Azure Stack Hub

To install OpenShift Container Platform on Microsoft Azure Stack Hub using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You manually create the **install-config.yaml** file, and then generate and customize the Kubernetes manifests and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

8.5.6.1. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

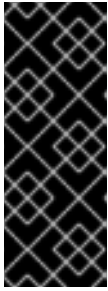
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

Make the following modifications for Azure Stack Hub:

- a. Set the **replicas** parameter to **0** for the **compute** pool:

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

- 1** Set to **0**.

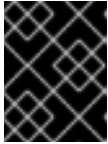
The compute machines will be provisioned manually later.

- b. Update the **platform.azure** section of the **install-config.yaml** file to configure your Azure Stack Hub configuration:

```
platform:
  azure:
    armEndpoint: <azurestack_arm_endpoint> 1
    baseDomainResourceGroupName: <resource_group> 2
    cloudName: AzureStackCloud 3
    region: <azurestack_region> 4
```

- 1** Specify the Azure Resource Manager endpoint of your Azure Stack Hub environment, like **https://management.local.azurestack.external**.
- 2** Specify the name of the resource group that contains the DNS zone for your base domain.
- 3** Specify the Azure Stack Hub environment, which is used to configure the Azure SDK with the appropriate Azure API endpoints.
- 4** Specify the name of your Azure Stack Hub region.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

8.5.6.2. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```

apiVersion: v1
baseDomain: example.com
controlPlane: 1
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 2
        diskType: premium_LRS
      replicas: 3
compute: 3
  - name: worker
    platform:
      azure:
        osDisk:
          diskSizeGB: 512 4
          diskType: premium_LRS
        replicas: 0
metadata:
  name: test-cluster 5
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 6
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint 7
    baseDomainResourceGroupName: resource_group 8
    region: azure_stack_local_region 9
    resourceGroupName: existing_resource_group 10
    outboundType: Loadbalancer
    cloudName: AzureStackCloud 11
pullSecret: '{"auths": ...}' 12

```

```
fips: false 13
additionalTrustBundle: | 14
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
sshKey: ssh-ed25519 AAAA... 15
```

- 1 3** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 2 4** You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.
- 5** Specify the name of the cluster.
- 6** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 7** Specify the Azure Resource Manager endpoint that your Azure Stack Hub operator provides.
- 8** Specify the name of the resource group that contains the DNS zone for your base domain.
- 9** Specify the name of your Azure Stack Hub local region.
- 10** Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
- 11** Specify the Azure Stack Hub environment as your target platform.
- 12** Specify the pull secret required to authenticate your cluster.
- 13** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 14** If your Azure Stack Hub environment uses an internal certificate authority (CA), add the necessary certificate bundle in **.pem** format.
- 15** You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

8.5.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.

- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

8.5.6.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure Stack Hub.

**NOTE**

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Export common variables found in the **install-config.yaml** to be used by the provided ARM templates:

```
$ export CLUSTER_NAME=<cluster_name> 1
$ export AZURE_REGION=<azure_region> 2
$ export SSH_KEY=<ssh_key> 3
$ export BASE_DOMAIN=<base_domain> 4
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> 5
```

- 1 The value of the **.metadata.name** attribute from the **install-config.yaml** file.
- 2 The region to deploy the cluster into. This is the value of the **.platform.azure.region** attribute from the **install-config.yaml** file.
- 3 The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the **.sshKey** attribute from the **install-config.yaml** file.
- 4 The base domain to deploy the cluster to. The base domain corresponds to the DNS zone that you created for your cluster. This is the value of the **.baseDomain** attribute from the **install-config.yaml** file.
- 5 The resource group where the DNS zone exists. This is the value of the **.platform.azure.baseDomainResourceGroupName** attribute from the **install-config.yaml** file.

For example:

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. Export the kubeadmin credentials:

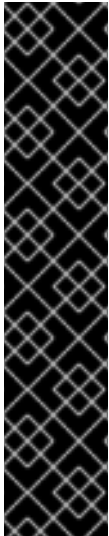
```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

8.5.6.5. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

- a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
 - c. Save and exit the file.
5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: ❶
    id: mycluster-100419-private-zone
  publicZone: ❷
    id: example.openshift.com
status: {}
```

- ❶ ❷ Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. Optional: If your Azure Stack Hub environment uses an internal certificate authority (CA), you must update the `.spec.trustedCA.name` field in the `<installation_directory>/manifests/cluster-proxy-01-config.yaml` file to use `user-ca-bundle`:

```
...
spec:
  trustedCA:
    name: user-ca-bundle
...
```

Later, you must update your bootstrap ignition to include the CA.

7. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

- a. Export the infrastructure ID by using the following command:

```
$ export INFRA_ID=<infra_id> ❶
```

- ❶ The OpenShift Container Platform cluster has been assigned an identifier (`INFRA_ID`) in the form of `<cluster_name>-<random_string>`. This will be used as the base name for most resources created using the provided ARM templates. This is the value of the `.status.infrastructureName` attribute from the `manifests/cluster-infrastructure-02-config.yml` file.

- b. Export the resource group by using the following command:

```
$ export RESOURCE_GROUP=<resource_group> 1
```

- 1 All resources created in this Azure deployment exists as part of a [resource group](#). The resource group name is also based on the **INFRA_ID**, in the form of **<cluster_name>-<random_string>-rg**. This is the value of the **.status.platformStatus.azure.resourceGroupName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

8. Manually create your cloud credentials.

- a. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

- b. Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=azure
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample **CredentialsRequest** object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-azure
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
```

- c. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object. The format for the secret data varies for each cloud provider.

Sample **secrets.yaml** file:

```

apiVersion: v1
kind: Secret
metadata:
  name: ${secret_name}
  namespace: ${secret_namespace}
stringData:
  azure_subscription_id: ${subscription_id}
  azure_client_id: ${app_id}
  azure_client_secret: ${client_secret}
  azure_tenant_id: ${tenant_id}
  azure_resource_prefix: ${cluster_name}
  azure_resourcegroup: ${resource_group}
  azure_region: ${azure_region}

```

9. Optional: If you manually created a cloud identity and access management (IAM) role, locate any **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation in the release image by running the following command:

```

$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --
credentials-requests --cloud=<platform_name>

```

Example output

```

0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade

```

IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- Delete all **CredentialsRequest** objects that have the **TechPreviewNoUpgrade** annotation.
10. Create a **cco-configmap.yaml** file in the manifests directory with the Cloud Credential Operator (CCO) disabled:

Sample ConfigMap object

```

apiVersion: v1
kind: ConfigMap
metadata:
  name: cloud-credential-operator-config
  namespace: openshift-cloud-credential-operator
annotations:

```

```

    release.openshift.io/create-only: "true"
  data:
    disabled: "true"

```

- To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```

.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign

```

Additional resources

- [Manually creating IAM](#)

8.5.6.6. Optional: Creating a separate **/var** partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- /var/lib/containers:** Holds container-related content that can grow as more images and containers are added to a system.
- /var/lib/etcd:** Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- /var:** Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
  partitions:
```

```

- label: var
  start_mib: <partition_start_offset> 2
  size_mib: <partition_size> 3
  number: 5
filesystems:
- device: /dev/disk/by-partlabel/var
  path: /var
  format: xfs
  mount_options: [defaults, prjquota] 4
  with_mount_unit: true

```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

8.5.7. Creating the Azure resource group

You must create a Microsoft Azure [resource group](#). This is used during the installation of your OpenShift Container Platform cluster on Azure Stack Hub.

Prerequisites

- Configure an Azure account.

- Generate the Ignition config files for your cluster.

Procedure

- Create the resource group in a supported Azure region:

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

8.5.8. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally. You must copy and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a storage container so they are accessible during deployment.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```



WARNING

The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your **CLUSTER_NAME** variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see [Resolve errors for storage account names](#) in the Azure documentation.

2. Export the storage account key as an environment variable:

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. Export the URL of the RHCOS VHD to an environment variable:

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r '.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Create the storage container for the VHD:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

5. Download the compressed RHCOS VHD file locally:

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

6. Decompress the VHD file.

**NOTE**

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. You can delete the VHD file after you upload it.

7. Copy the local VHD to a blob:

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -f rhcos-<rhcos_version>-azurestack.x86_64.vhd
```

8. Create a blob storage container and upload the generated **bootstrap.ign** file:

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

8.5.9. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, [Azure Stack Hub's datacenter DNS integration](#) is used, so you will create a DNS zone.

**NOTE**

The DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the DNS zone; be sure the installation config you generated earlier reflects that scenario.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

- Create the new DNS zone in the resource group exported in the **BASE_DOMAIN_RESOURCE_GROUP** environment variable:

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can skip this step if you are using a DNS zone that already exists.

You can learn more about [configuring a DNS zone in Azure Stack Hub](#) by visiting that section.

8.5.10. Creating a VNet in Azure Stack Hub

You must create a virtual network (VNet) in Microsoft Azure Stack Hub for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.



NOTE

If you do not use the provided ARM template to create your Azure Stack Hub infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the **ARM template for the VNet** section of this topic and save it as **01_vnet.json** in your cluster's installation directory. This template describes the VNet that your cluster requires.
2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
--template-file "<installation_directory>/01_vnet.json" \
--parameters baseName="${INFRA_ID}" 1
```

- 1** The base name to be used in resource names; this is usually the cluster's infrastructure ID.

8.5.10.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

Example 8.1. 01_vnet.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/azurestack/01_vnet.json

8.5.11. Deploying the RHCOS cluster image for the Azure Stack Hub infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure Stack Hub for your OpenShift Container Platform nodes.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.
- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the **ARM template for image storage** section of this topic and save it as **02_storage.json** in your cluster's installation directory. This template describes the image storage that your cluster requires.
2. Export the RHCOS VHD blob URL as a variable:

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. Deploy the cluster image:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
--template-file "<installation_directory>/02_storage.json" \
--parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
--parameters baseName="${INFRA_ID}" 2
```

1 The blob URL of the RHCOS VHD to be used to create master and worker machines.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

8.5.11.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

Example 8.2. 02_storage.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/azurestack/02_storage.json

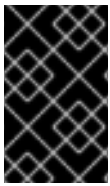
8.5.12. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

8.5.12.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 8.16. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 8.17. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 8.18. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

8.5.13. Creating networking and load balancing components in Azure Stack Hub

You must configure networking and load balancing in Microsoft Azure Stack Hub for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.

Load balancing requires the following DNS records:

- An **api** DNS record for the API public load balancer in the DNS zone.
- An **api-int** DNS record for the API internal load balancer in the DNS zone.



NOTE

If you do not use the provided ARM template to create your Azure Stack Hub infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure Stack Hub.

Procedure

1. Copy the template from the **ARM template for the network and load balancers** section of this topic and save it as **03_infra.json** in your cluster's installation directory. This template describes the networking and load balancing objects that your cluster requires.
2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters baseName="${INFRA_ID}" 1
```

- 1** The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Create an **api** DNS record and an **api-int** DNS record. When creating the API DNS records, the **\${BASE_DOMAIN_RESOURCE_GROUP}** variable must point to the resource group where the DNS zone exists.

- a. Export the following variable:

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

- b. Export the following variable:

```
$ export PRIVATE_IP=`az network lb frontend-ip show -g "$RESOURCE_GROUP" --lb-
name "${INFRA_ID}-internal" -n internal-lb-ip --query "privateIpAddress" -o tsv`
```

- c. Create the **api** DNS record in a new DNS zone:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

If you are adding the cluster to an existing DNS zone, you can create the **api** DNS record in it instead:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

- d. Create the **api-int** DNS record in a new DNS zone:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z "${CLUSTER_NAME}.${BASE_DOMAIN}" -n api-int -a ${PRIVATE_IP} --ttl 60
```

If you are adding the cluster to an existing DNS zone, you can create the **api-int** DNS record in it instead:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api-int.${CLUSTER_NAME} -a ${PRIVATE_IP} --ttl 60
```

8.5.13.1. ARM template for the network and load balancers

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

Example 8.3. 03_infra.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/azurestack/03_infra.json

8.5.14. Creating the bootstrap machine in Azure Stack Hub

You must create the bootstrap machine in Microsoft Azure Stack Hub to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.



NOTE

If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure Stack Hub.
- Create and configure networking and load balancers in Azure Stack Hub.
- Create control plane and compute roles.

Procedure

1. Copy the template from the **ARM template for the bootstrap machine** section of this topic and save it as **04_bootstrap.json** in your cluster's installation directory. This template describes the bootstrap machine that your cluster requires.
2. Export the bootstrap URL variable:

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. Export the bootstrap ignition variable:
 - a. If your environment uses a public certificate authority (CA), run this command:

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url
${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -
d '\n`
```

- b. If your environment uses an internal CA, you must add your PEM encoded bundle to the bootstrap ignition stub so that your bootstrap virtual machine can pull the bootstrap ignition from the storage account. Run the following commands, which assume your CA is in a file called **CA.pem**:

```
$ export CA="data:text/plain;charset=utf-8;base64,$(cat CA.pem |base64 |tr -d '\n')"
```

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url
"$BOOTSTRAP_URL" --arg cert "$CA" '{ignition:{version:$v,security:{tls:
{certificateAuthorities:[{source:$cert}]},config:{replace:{source:$url}}}}' | base64 | tr -d
'\n`
```

4. Create the deployment by using the **az** CLI:

```
$ az deployment group create --verbose -g ${RESOURCE_GROUP} \
--template-file "<installation_directory>/04_bootstrap.json" \
--parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
--parameters baseName="${INFRA_ID}" \ 2
--parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

- 1** The bootstrap Ignition content for the bootstrap cluster.
- 2** The base name to be used in resource names; this is usually the cluster's infrastructure ID.
- 3** The name of the storage account for your cluster.

8.5.14.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 8.4. 04_bootstrap.json ARM template

```
link:https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/azurestack/04_bootstrap.json[]
```

8.5.15. Creating the control plane machines in Azure Stack Hub

You must create the control plane machines in Microsoft Azure Stack Hub for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure Stack Hub.
- Create and configure networking and load balancers in Azure Stack Hub.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the **ARM template for control plane machines** section of this topic and save it as **05_masters.json** in your cluster's installation directory. This template describes the control plane machines that your cluster requires.
2. Export the following variable needed by the control plane machine deployment:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" \ 3
```

1 The Ignition content for the control plane nodes (also known as the master nodes).

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3 The name of the storage account for your cluster.

8.5.15.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 8.5. 05_masters.json ARM template

```
link:https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/azurestack/05\_masters.json]
```

8.5.16. Wait for bootstrap completion and remove bootstrap resources in Azure Stack Hub

After you create all of the required infrastructure in Microsoft Azure Stack Hub, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure Stack Hub.
- Create and configure networking and load balancers in Azure Stack Hub.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level info 2

```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```

$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip

```



NOTE

If you do not delete the bootstrap server, installation may not succeed due to API traffic being routed to the bootstrap server.

8.5.17. Creating additional worker machines in Azure Stack Hub

You can create worker machines in Microsoft Azure Stack Hub for your cluster to use by launching

individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type **06_workers.json** in the file.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure Stack Hub.
- Create and configure networking and load balancers in Azure Stack Hub.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster's installation directory. This template describes the worker machines that your cluster requires.
2. Export the following variable needed by the worker machine deployment:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \  
  --template-file "<installation_directory>/06_workers.json" \  
  --parameters workerIgnition="${WORKER_IGNITION}" ① \  
  --parameters baseName="${INFRA_ID}" ② \  
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" ③
```

- ① The Ignition content for the worker nodes.
- ② The base name to be used in resource names; this is usually the cluster's infrastructure ID.
- ③ The name of the storage account for your cluster.

8.5.17.1. ARM template for worker machines

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 8.6. 06_workers.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/azurestack/06_workers.json

8.5.18. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

8.5.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The

kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

8.5.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 63m  v1.25.0
master-1  Ready   master 63m  v1.25.0
master-2  Ready   master 64m  v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME          AGE   REQUESTOR                                     CONDITION
csr-8b2br     15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps     15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

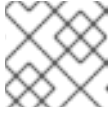
- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
```

```

master-2 Ready   master 74m v1.25.0
worker-0 Ready   worker 11m v1.25.0
worker-1 Ready   worker 11m v1.25.0

```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

8.5.21. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Microsoft Azure Stack Hub by using infrastructure that you provisioned.
- Install the OpenShift CLI (**oc**).
- Install or update the [Azure CLI](#).

Procedure

1. Confirm the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```

NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
router-default LoadBalancer  172.30.20.10  35.130.120.110 80:32288/TCP,443:31215/TCP 20

```

2. Export the Ingress router IP as a variable:

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

3. Add a ***.apps** record to the DNS zone.

- a. If you are adding this cluster to a new DNS zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

b. If you are adding this cluster to an already existing DNS zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

Example output

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

8.5.22. Completing an Azure Stack Hub installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure Stack Hub user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure Stack Hub infrastructure.
- Install the **oc** CLI and log in.

Procedure

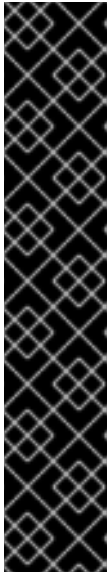
- Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

8.6. UNINSTALLING A CLUSTER ON AZURE STACK HUB

You can remove a cluster that you deployed to Azure Stack Hub.

8.6.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

While you can uninstall the cluster using the copy of the installation program that was used to deploy it, using OpenShift Container Platform version 4.13 or later is recommended.

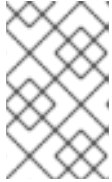
The removal of service principals is dependent on the Microsoft Azure AD Graph API. Using version 4.13 or later of the installation program ensures that service principals are removed without the need for manual intervention, if and when Microsoft decides to [retire](#) the Azure AD Graph API.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$. /openshift-install destroy cluster \  
--dir <installation_directory> --log-level info 1 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 9. INSTALLING ON GCP

9.1. PREPARING TO INSTALL ON GCP

9.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

9.1.2. Requirements for installing OpenShift Container Platform on GCP

Before installing OpenShift Container Platform on Google Cloud Platform (GCP), you must create a service account and configure a GCP project. See [Configuring a GCP project](#) for details about creating a project, enabling API services, configuring DNS, GCP account limits, and supported GCP regions.

If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, see [Manually creating IAM for GCP](#) for other options.

9.1.3. Choosing a method to install OpenShift Container Platform on GCP

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

9.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on GCP infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- [Installing a cluster quickly on GCP](#) You can install OpenShift Container Platform on GCP infrastructure that is provisioned by the OpenShift Container Platform installation program. You can install a cluster quickly by using the default configuration options.
- [Installing a customized cluster on GCP](#) You can install a customized cluster on GCP infrastructure that the installation program provisions. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).
- [Installing a cluster on GCP with network customizations](#) You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- [Installing a cluster on GCP in a restricted network](#) You can install OpenShift Container Platform on GCP on installer-provisioned infrastructure by using an internal mirror of the installation release content. You can use this method to install a cluster that does not require an

active internet connection to obtain the software components. While you can install OpenShift Container Platform by using the mirrored content, your cluster still requires internet access to use the GCP APIs.

- **Installing a cluster into an existing Virtual Private Cloud** You can install OpenShift Container Platform on an existing GCP Virtual Private Cloud (VPC). You can use this installation method if you have constraints set by the guidelines of your company, such as limits on creating new accounts or infrastructure.
- **Installing a private cluster on an existing VPC** You can install a private cluster on an existing GCP VPC. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.

9.1.3.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on GCP infrastructure that you provision, by using one of the following methods:

- **Installing a cluster on GCP with user-provisioned infrastructure** You can install OpenShift Container Platform on GCP infrastructure that you provide. You can use the provided Deployment Manager templates to assist with the installation.
- **Installing a cluster with shared VPC on user-provisioned infrastructure in GCP** You can use the provided Deployment Manager templates to create GCP resources in a shared VPC infrastructure.
- **Installing a cluster on GCP in a restricted network with user-provisioned infrastructure** You can install OpenShift Container Platform on GCP in a restricted network with user-provisioned infrastructure. By creating an internal mirror of the installation release content, you can install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

9.1.4. Next steps

- [Configuring a GCP project](#)

9.2. CONFIGURING A GCP PROJECT

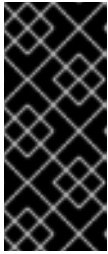
Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

9.2.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See [Creating and Managing Projects](#) in the GCP documentation.



IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the **api-int.<cluster_name>.<base_domain>** URL; the Premium Tier is required for internal load balancing.

9.2.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. You may also enable optional API services which are not required for installation. See [Enabling services](#) in the GCP documentation.

Table 9.1. Required API services

API service	Console service name
Compute Engine API	compute.googleapis.com
Cloud Resource Manager API	cloudresourcemanager.googleapis.com
Google DNS API	dns.googleapis.com
IAM Service Account Credentials API	iamcredentials.googleapis.com
Identity and Access Management (IAM) API	iam.googleapis.com
Service Usage API	serviceusage.googleapis.com

Table 9.2. Optional API services

API service	Console service name
Google Cloud APIs	cloudapis.googleapis.com
Service Management API	servicemanagement.googleapis.com
Google Cloud Storage JSON API	storage-api.googleapis.com

API service	Console service name
Cloud Storage	storage-component.googleapis.com

9.2.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.



NOTE

If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](#).

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](#) in the GCP documentation. Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](#) in the GCP documentation. You typically have four name servers.
4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](#).
5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](#) in the GCP documentation.
6. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain. This process might include a request to your company's IT department or the division that controls the root domain and DNS services for your company.

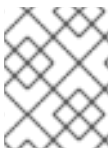
9.2.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default [Quotas](#) do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

Table 9.3. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	6	1
Firewall rules	Compute	Global	11	1
Forwarding rules	Compute	Global	2	0
In-use global IP addresses	Compute	Global	4	1
Health checks	Compute	Global	3	0
Images	Compute	Global	1	0
Networks	Compute	Global	2	0
Static IP addresses	Compute	Region	4	1
Routers	Compute	Global	1	0
Routes	Compute	Global	2	0
Subnetworks	Compute	Global	2	0
Target pools	Compute	Global	3	0
CPUs	Compute	Region	28	4
Persistent disk SSD (GB)	Compute	Region	896	128



NOTE

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- **asia-east2**
- **asia-northeast2**

- **asia-south1**
- **australia-southeast1**
- **europa-north1**
- **europa-west2**
- **europa-west3**
- **europa-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

You can increase resource quotas from the [GCP console](#), but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

9.2.5. Creating a service account in GCP

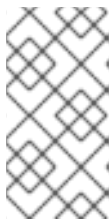
OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See [Creating a service account](#) in the GCP documentation.
2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See [Granting roles to a service account for specific resources](#).



NOTE

While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. You can create the service account key in JSON format, or attach the service account to a GCP virtual machine. See [Creating service account keys](#) and [Creating and enabling service accounts for instances](#) in the GCP documentation.
You must have a service account key or a virtual machine with an attached service account to create the cluster.

**NOTE**

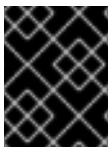
If you use a virtual machine with an attached service account to create your cluster, you must set **credentialsMode: Manual** in the **install-config.yaml** file before installation.

Additional resources

- See [Manually creating IAM](#) for more details about using manual credentials mode.

9.2.5.1. Required GCP roles

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. If the security policies for your organization require a more restrictive set of permissions, you can create a service account with the following permissions.

**IMPORTANT**

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions.

If you deploy your cluster into an existing virtual private cloud (VPC), the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- IAM Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Required roles for using passthrough credentials mode

- Compute Load Balancer Admin
- IAM Role Viewer

The roles are applied to the service accounts that the control plane and compute machines use:

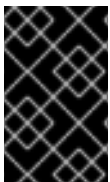
Table 9.4. GCP service account permissions

Account	Roles
Control Plane	roles/compute.instanceAdmin
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	roles/compute.viewer
	roles/storage.admin

9.2.5.2. Required GCP permissions for installer-provisioned infrastructure

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform.

If the security policies for your organization require a more restrictive set of permissions, you can create [custom roles](#) with the necessary permissions. The following permissions are required for the installer-provisioned infrastructure for creating and deleting the OpenShift Container Platform cluster.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions. For more information, see "Required roles for using passthrough credentials mode" in the "Required GCP roles" section.

Example 9.1. Required permissions for creating network resources

- **compute.addresses.create**
- **compute.addresses.createInternal**
- **compute.addresses.delete**
- **compute.addresses.get**
- **compute.addresses.list**
- **compute.addresses.use**
- **compute.addresses.useInternal**
- **compute.firewalls.create**
- **compute.firewalls.delete**

- `compute.firewalls.get`
- `compute.firewalls.list`
- `compute.forwardingRules.create`
- `compute.forwardingRules.get`
- `compute.forwardingRules.list`
- `compute.forwardingRules.setLabels`
- `compute.networks.create`
- `compute.networks.get`
- `compute.networks.list`
- `compute.networks.updatePolicy`
- `compute.routers.create`
- `compute.routers.get`
- `compute.routers.list`
- `compute.routers.update`
- `compute.routes.list`
- `compute.subnetworks.create`
- `compute.subnetworks.get`
- `compute.subnetworks.list`
- `compute.subnetworks.use`
- `compute.subnetworks.useExternallp`

Example 9.2. Required permissions for creating load balancer resources

- `compute.regionBackendServices.create`
- `compute.regionBackendServices.get`
- `compute.regionBackendServices.list`
- `compute.regionBackendServices.update`
- `compute.regionBackendServices.use`
- `compute.targetPools.addInstance`
- `compute.targetPools.create`
- `compute.targetPools.get`

- `compute.targetPools.list`
- `compute.targetPools.removeInstance`
- `compute.targetPools.use`

Example 9.3. Required permissions for creating DNS resources

- `dns.changes.create`
- `dns.changes.get`
- `dns.managedZones.create`
- `dns.managedZones.get`
- `dns.managedZones.list`
- `dns.networks.bindPrivateDNSZone`
- `dns.resourceRecordSets.create`
- `dns.resourceRecordSets.list`

Example 9.4. Required permissions for creating Service Account resources

- `iam.serviceAccountKeys.create`
- `iam.serviceAccountKeys.delete`
- `iam.serviceAccountKeys.get`
- `iam.serviceAccountKeys.list`
- `iam.serviceAccounts.actAs`
- `iam.serviceAccounts.create`
- `iam.serviceAccounts.delete`
- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.get`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.5. Required permissions for creating compute resources

- `compute.disks.create`

- **compute.disks.get**
- **compute.disks.list**
- **compute.instanceGroups.create**
- **compute.instanceGroups.delete**
- **compute.instanceGroups.get**
- **compute.instanceGroups.list**
- **compute.instanceGroups.update**
- **compute.instanceGroups.use**
- **compute.instances.create**
- **compute.instances.delete**
- **compute.instances.get**
- **compute.instances.list**
- **compute.instances.setLabels**
- **compute.instances.setMetadata**
- **compute.instances.setServiceAccount**
- **compute.instances.setTags**
- **compute.instances.use**
- **compute.machineTypes.get**
- **compute.machineTypes.list**

Example 9.6. Required for creating storage resources

- **storage.buckets.create**
- **storage.buckets.delete**
- **storage.buckets.get**
- **storage.buckets.list**
- **storage.objects.create**
- **storage.objects.delete**
- **storage.objects.get**
- **storage.objects.list**

Example 9.7. Required permissions for creating health check resources

- **compute.healthChecks.create**
- **compute.healthChecks.get**
- **compute.healthChecks.list**
- **compute.healthChecks.useReadOnly**
- **compute.httpHealthChecks.create**
- **compute.httpHealthChecks.get**
- **compute.httpHealthChecks.list**
- **compute.httpHealthChecks.useReadOnly**

Example 9.8. Required permissions to get GCP zone and region related information

- **compute.globalOperations.get**
- **compute.regionOperations.get**
- **compute.regions.list**
- **compute.zoneOperations.get**
- **compute.zones.get**
- **compute.zones.list**

Example 9.9. Required permissions for checking services and quotas

- **monitoring.timeSeries.list**
- **serviceusage.quotas.get**
- **serviceusage.services.list**

Example 9.10. Required IAM permissions for installation

- **iam.roles.get**

Example 9.11. Optional Images permissions for installation

- **compute.images.list**

Example 9.12. Optional permission for running gather bootstrap

- **compute.instances.getSerialPortOutput**

Example 9.13. Required permissions for deleting network resources

- **compute.addresses.delete**
- **compute.addresses.deleteInternal**
- **compute.addresses.list**
- **compute.firewalls.delete**
- **compute.firewalls.list**
- **compute.forwardingRules.delete**
- **compute.forwardingRules.list**
- **compute.networks.delete**
- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.delete**
- **compute.routers.list**
- **compute.routes.list**
- **compute.subnetworks.delete**
- **compute.subnetworks.list**

Example 9.14. Required permissions for deleting load balancer resources

- **compute.regionBackendServices.delete**
- **compute.regionBackendServices.list**
- **compute.targetPools.delete**
- **compute.targetPools.list**

Example 9.15. Required permissions for deleting DNS resources

- **dns.changes.create**
- **dns.managedZones.delete**
- **dns.managedZones.get**
- **dns.managedZones.list**

- `dns.resourceRecordSets.delete`
- `dns.resourceRecordSets.list`

Example 9.16. Required permissions for deleting Service Account resources

- `iam.serviceAccounts.delete`
- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.17. Required permissions for deleting compute resources

- `compute.disks.delete`
- `compute.disks.list`
- `compute.instanceGroups.delete`
- `compute.instanceGroups.list`
- `compute.instances.delete`
- `compute.instances.list`
- `compute.instances.stop`
- `compute.machineTypes.list`

Example 9.18. Required for deleting storage resources

- `storage.buckets.delete`
- `storage.buckets.getIamPolicy`
- `storage.buckets.list`
- `storage.objects.delete`
- `storage.objects.list`

Example 9.19. Required permissions for deleting health check resources

- `compute.healthChecks.delete`
- `compute.healthChecks.list`

- **compute.httpHealthChecks.delete**
- **compute.httpHealthChecks.list**

Example 9.20. Required Images permissions for deletion

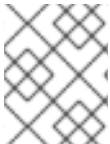
- **compute.images.list**

9.2.6. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **australia-southeast1** (Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europa-central2** (Warsaw, Poland)
- **europa-north1** (Hamina, Finland)
- **europa-southwest1** (Madrid, Spain)
- **europa-west1** (St. Ghislain, Belgium)
- **europa-west2** (London, England, UK)
- **europa-west3** (Frankfurt, Germany)
- **europa-west4** (Eemshaven, Netherlands)
- **europa-west6** (Zürich, Switzerland)
- **europa-west8** (Milan, Italy)
- **europa-west9** (Paris, France)

- **europa-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)



NOTE

To determine which machine type instances are available by region and zone, see the Google [documentation](#).

9.2.7. Next steps

- Install an OpenShift Container Platform cluster on GCP. You can [install a customized cluster](#) or [quickly install a cluster](#) with default options.

9.3. MANUALLY CREATING IAM FOR GCP

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

9.3.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the **credentialsMode** parameter in the **install-config.yaml** file.

If you prefer not to store an administrator-level credential secret in the cluster **kube-system** project, you can choose one of the following options when installing OpenShift Container Platform:

- **Use manual mode with GCP Workload Identity**

You can use the CCO utility (**ccoctl**) to configure the cluster to use manual mode with GCP Workload Identity. When the CCO utility is used to configure the cluster for GCP Workload Identity, it signs service account tokens that provide short-term, limited-privilege security credentials to components.



NOTE

This credentials strategy is supported for only new OpenShift Container Platform clusters and must be configured during installation. You cannot reconfigure an existing cluster that uses a different credentials strategy to use this feature.

- **Manage cloud credentials manually.**

You can set the **credentialsMode** parameter for the CCO to **Manual** to manage cloud credentials manually. Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

- **Remove the administrator-level credential secret after installing OpenShift Container Platform with mint mode:**

If you are using the CCO with the **credentialsMode** parameter set to **Mint**, you can remove or rotate the administrator-level credential after installing OpenShift Container Platform. Mint mode is the default configuration for the CCO. This option requires the presence of the administrator-level credential during an installation. The administrator-level credential is used during the installation to mint other credentials with some permissions granted. The original credential secret is not stored in the cluster permanently.



NOTE

Prior to a non z-stream upgrade, you must reinstate the credential secret with the administrator-level credential. If the credential is not present, the upgrade might be blocked.

Additional resources

- [Using manual mode with GCP Workload Identity](#)
- [Rotating or removing cloud provider credentials](#)

For a detailed description of all available CCO credential modes and their supported platforms, see [About the Cloud Credential Operator](#).

9.3.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. Change to the directory that contains the installation program and create the **install-config.yaml** file by running the following command:

```
$ openshift-install create install-config --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

2. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
...
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

3. Generate the manifests by running the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

4. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use by running the following command:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

5. Locate all **CredentialsRequest** objects in this release image that target the cloud you are deploying on by running the following command:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 \
  --credentials-requests \
  --cloud=gcp
```

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
```

```

kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
      - roles/storage.admin
      - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...

```

6. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```

apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component-credentials-request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    ...
  secretRef:
    name: <component-secret>
    namespace: <component-namespace>
  ...

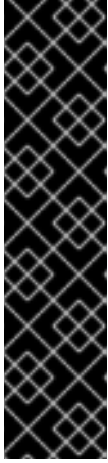
```

Sample Secret object

```

apiVersion: v1
kind: Secret
metadata:
  name: <component-secret>
  namespace: <component-namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>

```



IMPORTANT

The release image includes **CredentialsRequest** objects for Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set. You can identify these objects by their use of the **release.openshift.io/feature-set: TechPreviewNoUpgrade** annotation.

- If you are not using any of these features, do not create secrets for these objects. Creating secrets for Technology Preview features that you are not using can cause the installation to fail.
- If you are using any of these features, you must create secrets for the corresponding objects.

- To find **CredentialsRequest** objects with the **TechPreviewNoUpgrade** annotation, run the following command:

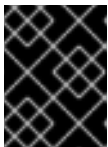
```
$ grep "release.openshift.io/feature-set" *
```

Example output

```
0000_30_capi-operator_00_credentials-request.yaml: release.openshift.io/feature-set:
TechPreviewNoUpgrade
```

7. From the directory that contains the installation program, proceed with your cluster creation:

```
$ openshift-install create cluster --dir <installation_directory>
```



IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- [Updating a cluster using the web console](#)
- [Updating a cluster using the CLI](#)

9.3.3. Mint mode

Mint mode is the default Cloud Credential Operator (CCO) credentials mode for OpenShift Container Platform on platforms that support it. In this mode, the CCO uses the provided administrator-level cloud credential to run the cluster. Mint mode is supported for AWS and GCP.

In mint mode, the **admin** credential is stored in the **kube-system** namespace and then used by the CCO to process the **CredentialsRequest** objects in the cluster and create users for each with specific permissions.

The benefits of mint mode include:

- Each cluster component has only the permissions it requires

- Automatic, on-going reconciliation for cloud credentials, including additional credentials or permissions that might be required for upgrades

One drawback is that mint mode requires **admin** credential storage in a cluster **kube-system** secret.

9.3.4. Mint mode with removal or rotation of the administrator-level credential

Currently, this mode is only supported on AWS and GCP.

In this mode, a user installs OpenShift Container Platform with an administrator-level credential just like the normal mint mode. However, this process removes the administrator-level credential secret from the cluster post-installation.

The administrator can have the Cloud Credential Operator make its own request for a read-only credential that allows it to verify if all **CredentialsRequest** objects have their required permissions, thus the administrator-level credential is not required unless something needs to be changed. After the associated credential is removed, it can be deleted or deactivated on the underlying cloud, if desired.



NOTE

Prior to a non z-stream upgrade, you must reinstate the credential secret with the administrator-level credential. If the credential is not present, the upgrade might be blocked.

The administrator-level credential is not stored in the cluster permanently.

Following these steps still requires the administrator-level credential in the cluster for brief periods of time. It also requires manually re-instating the secret with administrator-level credentials for each upgrade.

9.3.5. Next steps

- Install an OpenShift Container Platform cluster:
 - [Installing a cluster quickly on GCP](#) with default options on installer-provisioned infrastructure
 - [Install a cluster with cloud customizations on installer-provisioned infrastructure](#)
 - [Install a cluster with network customizations on installer-provisioned infrastructure](#)

9.4. INSTALLING A CLUSTER QUICKLY ON GCP

In OpenShift Container Platform version 4.12, you can install a cluster on Google Cloud Platform (GCP) that uses the default configuration options.

9.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

9.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.4.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

■

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.4.4. Obtaining the installation program

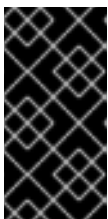
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

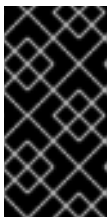
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.4.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GCPLOUD_KEYFILE_JSON** environment variables
 - The `~/.gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want

to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

3. Provide values at the prompts:

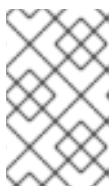
- a. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- b. Select **gcp** as the platform to target.
- c. If you have not configured the service account key for your GCP account on your host, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
- d. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
- e. Select the region to deploy the cluster to.
- f. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
- g. Enter a descriptive name for your cluster. If you provide a name that is longer than 6 characters, only the first 6 characters will be used in the infrastructure ID that is generated from the cluster name.
- h. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

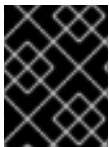
4. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
- If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.

- Credential information also outputs to `<installation_directory>/openshift_install.log`.

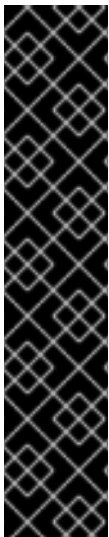


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

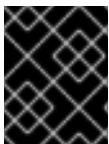


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.4.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.

3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.4.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.4.8. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use [subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.4.9. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).

9.5. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on infrastructure that the installation program provisions on Google Cloud Platform (GCP). To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

9.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

9.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.5.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

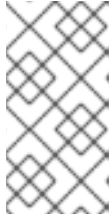
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1** Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

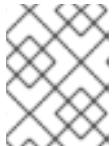
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.5.4. Obtaining the installation program

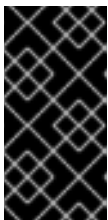
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

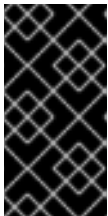
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.5.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.

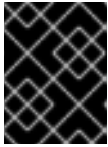


NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
- iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
- iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
- v. Select the region to deploy the cluster to.
- vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
- vii. Enter a descriptive name for your cluster.

- viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

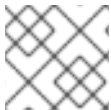


IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.5.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.5.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.5. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .

Parameter	Description	Values
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.5.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.6. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .


Parameter	Description	Values
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


9.5.5.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 9.7. Optional parameters

Parameter	Description	Values
additionalTrustBundle	<p>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</p>	String
capabilities	<p>Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i>.</p>	String array

Parameter	Description	Values
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled

Parameter	Description	Values
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
<p>fips</p>	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 592 1393" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	<p>false or true</p>

Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p>  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

9.5.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.8. Additional GCP parameters

Parameter	Description	Values
platform.gcp.network	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.

Parameter	Description	Values
<code>platform.gcp.networkProjectID</code>	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.
<code>platform.gcp.projectID</code>	The name of the GCP project where the installation program installs the cluster.	String.
<code>platform.gcp.region</code>	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
<code>platform.gcp.controlPlaneSubnet</code>	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
<code>platform.gcp.computeSubnet</code>	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
<code>platform.gcp.createFirewallRules</code>	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
<code>platform.gcp.publicDNSZone.project</code>	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.

Parameter	Description	Values
<code>platform.gcp.publicDNSZoneId</code>	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.
<code>platform.gcp.privateDNSZoneProject</code>	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the host project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the private DNS zone.
<code>platform.gcp.privateDNSZoneId</code>	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
<code>platform.gcp.licenses</code>	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>The licenses parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
<code>platform.gcp.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImageProject</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImageName</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImageProject</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing</code>	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.location</code>	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID	The ID of the project in which the KMS key ring exists. This value defaults to the value of the platform.gcp.projectID parameter if it is not set.	The GCP project ID.
platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.ServiceAccount	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <service_account_name>@<project_id>.iam.gserviceaccount.com .
controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKeyLocation</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDiskEncryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDiskEncryptionKeyServiceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDiskSizeGB</code>	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
<code>controlPlane.platform.gcp.osDiskType</code>	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
<code>controlPlane.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
<code>controlPlane.platform.gcp.osImage.name</code>	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use <code>controlPlane.platform.gcp.osImage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>controlPlane.platform.gcp.tags</code>	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
<code>controlPlane.platform.gcp.type</code>	The GCP machine type for control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
controlPlane.platform.gcp.zones	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
compute.platform.gcp.osDisk.encryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyLocation	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.osDisk.diskSizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.osDisk.diskType</code>	The GCP disk type for compute machines.	Either the default pd-ssd or the pd-standard disk type.
<code>compute.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

Parameter	Description	Values
compute.platform.gcp.osimage.name	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use compute.platform.gcp.osimage.project , this field is required.	String. The name of the RHCOS image.
compute.platform.gcp.tags	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.tags parameter for compute machines.	One or more strings, for example compute-network-tag1 .
compute.platform.gcp.type	The GCP machine type for compute machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.type parameter.	The GCP machine type, for example n1-standard-4 .
compute.platform.gcp.zones	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

9.5.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.9. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.5.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.21. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.5.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

As part of the installation process, you specify the custom machine type in the **install-config.yaml** file.

Sample install-config.yaml file with a custom machine type

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.5.5.5. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
```

```

- us-central1-c
osDisk:
  diskType: pd-ssd
  diskSizeGB: 1024
  encryptionKey: 5
  kmsKey:
    name: worker-key
    keyRing: test-machine-keys
    location: global
    projectID: project-id
  tags: 6
- control-plane-tag1
- control-plane-tag2
osImage: 7
  project: example-project-name
  name: example-image-name
replicas: 3
compute: 8 9
- hyperthreading: Enabled 10
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 11
        kmsKey:
          name: worker-key
          keyRing: test-machine-keys
          location: global
          projectID: project-id
      tags: 12
        - compute-tag1
        - compute-tag2
      osImage: 13
        project: example-project-name
        name: example-image-name
      replicas: 3
  metadata:
    name: test-cluster 14
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 15
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    gcp:

```

```

projectID: openshift-production 16
region: us-central1 17
defaultMachinePlatform:
  tags: 18
  - global-tag1
  - global-tag2
  osImage: 19
    project: example-project-name
    name: example-image-name
pullSecret: '{"auths": ...}' 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22

```

1 14 16 17 20 Required. The installation program prompts you for this value.

2 8 If you do not provide these parameters and values, the installation program provides the default value.

3 9 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 10 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

5 11 Optional: The custom encryption key section to encrypt both virtual machines and persistent volumes. Your default compute service account must have the permissions granted to use your KMS key and have the correct IAM role assigned. The default service account name follows the **service-<project_number>@compute-system.iam.gserviceaccount.com** pattern. For more information about granting the correct permissions for your service account, see "Machine management" → "Creating compute machine sets" → "Creating a compute machine set on GCP".

6 12 18 Optional: A set of network tags to apply to the control plane or compute machine sets. The **platform.gcp.defaultMachinePlatform.tags** parameter will apply to both control plane and compute machines. If the **compute.platform.gcp.tags** or **controlPlane.platform.gcp.tags** parameters are set, they override the **platform.gcp.defaultMachinePlatform.tags** parameter.

7 13 19 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image for the installation program to use to boot control plane and compute machines. The **project** and **name** parameters under **platform.gcp.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the **project** and **name** parameters under **controlPlane.platform.gcp.osImage** or **compute.platform.gcp.osImage** are set, they override the **platform.gcp.defaultMachinePlatform.osImage** parameters.

- 15 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 21 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS validated or Modules In Progress cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 22 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

Additional resources

- [Enabling customer-managed encryption keys for a compute machine set](#)

9.5.5.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



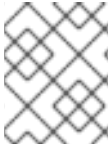
NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.5.6. Using the GCP Marketplace offering

Using the GCP Marketplace offering lets you deploy an OpenShift Container Platform cluster, which is billed on pay-per-use basis (hourly, per core) through GCP, while still being supported directly by Red Hat.

By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to deploy compute machines. To deploy an OpenShift Container Platform cluster using an RHCOS image from the GCP Marketplace, override the default behavior by modifying the **install-config.yaml** file to reference the location of GCP Marketplace offer.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

1. Edit the **compute.platform.gcp.osImage** parameters to specify the location of the GCP Marketplace image:

- Set the **project** parameter to **redhat-marketplace-public**.
- Set the **name** parameter to one of the following offerings:

OpenShift Container Platform

redhat-coreos-ocp-48-x86-64-202210040145

OpenShift Platform Plus

redhat-coreos-opp-48-x86-64-202206140145

OpenShift Kubernetes Engine

redhat-coreos-oke-48-x86-64-202206140145

2. Save the file and reference it when deploying the cluster.

Sample **install-config.yaml** file that specifies a GCP Marketplace image for compute machines

```
apiVersion: v1
baseDomain: example.com
controlPlane:
# ...
compute:
  platform:
    gcp:
      osImage:
```

```
project: redhat-marketplace-public
name: redhat-coreos-ocp-48-x86-64-202210040145
```

```
# ...
```

9.5.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GCLOUD_KEYFILE_JSON** environment variables
 - The `~/.gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized `./install-config.yaml` file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

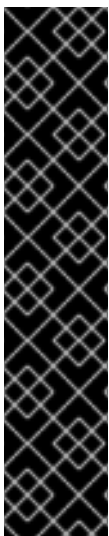


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.5.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

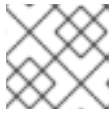
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.5.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.5.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.5.11. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).

9.6. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Google Cloud Platform (GCP). By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

You must set most of the network configuration parameters during installation, and you can modify only `kubeProxy` configuration parameters in a running cluster.

9.6.1. Prerequisites

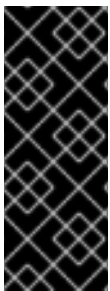
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

9.6.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.6.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.6.4. Obtaining the installation program

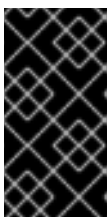
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.6.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

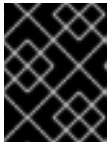
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
 - iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
 - iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
 - v. Select the region to deploy the cluster to.
 - vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - vii. Enter a descriptive name for your cluster.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.6.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.6.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.10. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.6.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.11. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


9.6.5.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 9.12. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 593 1393" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1442 593 1637" style="background-color: #f0f0f0; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...


9.6.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.13. Additional GCP parameters

Parameter	Description	Values
platform.gcp.network	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.

Parameter	Description	Values
<code>platform.gcp.networkProjectID</code>	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.
<code>platform.gcp.projectID</code>	The name of the GCP project where the installation program installs the cluster.	String.
<code>platform.gcp.region</code>	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
<code>platform.gcp.controlPlaneSubnet</code>	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
<code>platform.gcp.computeSubnet</code>	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
<code>platform.gcp.createFirewallRules</code>	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
<code>platform.gcp.publicDNSZone.project</code>	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.

Parameter	Description	Values
platform.gcp.publicDNSZoneId	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.
platform.gcp.privateDNSZoneProject	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the host project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the private DNS zone.
platform.gcp.privateDNSZoneId	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
platform.gcp.licenses	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>The licenses parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
platform.gcp.defaultMachinePlatform.zones	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImageProject</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImageName</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImageProject</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing</code>	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.location</code>	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID</code>	The ID of the project in which the KMS key ring exists. This value defaults to the value of the <code>platform.gcp.projectID</code> parameter if it is not set.	The GCP project ID.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.ServiceAccount</code>	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name</code>	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDiskEncryptionKey.location</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDiskEncryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDiskEncryptionKeyServiceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDiskSize</code> GB	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
<code>controlPlane.platform.gcp.osDiskType</code>	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
<code>controlPlane.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
<code>controlPlane.platform.gcp.osImage.name</code>	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use <code>controlPlane.platform.gcp.osImage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>controlPlane.platform.gcp.tags</code>	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
<code>controlPlane.platform.gcp.type</code>	The GCP machine type for control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
controlPlane.platform.gcp.zones	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
compute.platform.gcp.osDisk.encryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
compute.platform.gcp.osDisk.encryptionKey.location	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.osDisk.sizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.osDisk.type</code>	The GCP disk type for compute machines.	Either the default pd-ssd or the pd-standard disk type.
<code>compute.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

Parameter	Description	Values
compute.platform.gcp.osimage.name	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use compute.platform.gcp.osimage.project , this field is required.	String. The name of the RHCOS image.
compute.platform.gcp.tags	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.tags parameter for compute machines.	One or more strings, for example compute-network-tag1 .
compute.platform.gcp.type	The GCP machine type for compute machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.type parameter.	The GCP machine type, for example n1-standard-4 .
compute.platform.gcp.zones	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

9.6.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.14. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.6.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.22. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.6.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

As part of the installation process, you specify the custom machine type in the **install-config.yaml** file.

Sample install-config.yaml file with a custom machine type

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.6.5.5. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
    osDisk:
      diskType: pd-ssd
      diskSizeGB: 1024
```

```

    encryptionKey: 5
      kmsKey:
        name: worker-key
        keyRing: test-machine-keys
        location: global
        projectID: project-id
    tags: 6
      - control-plane-tag1
      - control-plane-tag2
    osImage: 7
      project: example-project-name
      name: example-image-name
  replicas: 3
compute: 8 9
- hyperthreading: Enabled 10
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 11
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
      tags: 12
        - compute-tag1
        - compute-tag2
      osImage: 13
        project: example-project-name
        name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 14
networking: 15
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 16
  serviceNetwork:
    - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 17
    region: us-central1 18
  defaultMachinePlatform:

```

```

tags: 19
- global-tag1
- global-tag2
osImage: 20
  project: example-project-name
  name: example-image-name
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23

```

1 14 17 18 21 Required. The installation program prompts you for this value.

2 8 15 If you do not provide these parameters and values, the installation program provides the default value.

3 9 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 10 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

5 11 Optional: The custom encryption key section to encrypt both virtual machines and persistent volumes. Your default compute service account must have the permissions granted to use your KMS key and have the correct IAM role assigned. The default service account name follows the **service-`<project_number>`@compute-system.iam.gserviceaccount.com** pattern. For more information about granting the correct permissions for your service account, see "Machine management" → "Creating compute machine sets" → "Creating a compute machine set on GCP".

6 12 19 Optional: A set of network tags to apply to the control plane or compute machine sets. The **platform.gcp.defaultMachinePlatform.tags** parameter will apply to both control plane and compute machines. If the **compute.platform.gcp.tags** or **controlPlane.platform.gcp.tags** parameters are set, they override the **platform.gcp.defaultMachinePlatform.tags** parameter.

7 13 20 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image for the installation program to use to boot control plane and compute machines. The **project** and **name** parameters under **platform.gcp.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the **project** and **name** parameters under **controlPlane.platform.gcp.osImage** or **compute.platform.gcp.osImage** are set, they override the **platform.gcp.defaultMachinePlatform.osImage** parameters.

16 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

22 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is

**IMPORTANT**

The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

23

You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

Additional resources

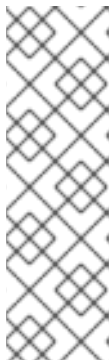
- [Enabling customer-managed encryption keys for a compute machine set](#)

9.6.5.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
```

```

httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

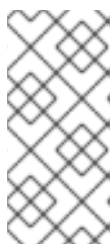
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.6.6. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

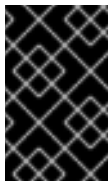
- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

9.6.7. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

9.6.8. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

9.6.8.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 9.15. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 9.16. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 9.17. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 9.18. `ovnKubernetesConfig` object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 9.19. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 9.20. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

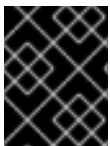
The values for the **kubeProxyConfig** object are defined in the following table:

Table 9.21. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

9.6.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

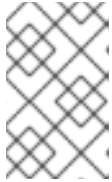
1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or

GCPLOUD_KEYFILE_JSON environment variables

- The `~/gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized `./install-config.yaml` file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
```

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.6.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

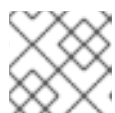
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.6.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.6.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.6.13. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

9.7. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.12, you can install a cluster on Google Cloud Platform (GCP) in a restricted network by creating an internal mirror of the installation release content on an existing Google Virtual Private Cloud (VPC).

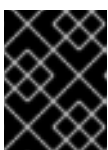


IMPORTANT

You can install an OpenShift Container Platform cluster by using mirrored installation release content, but your cluster will require internet access to use the GCP APIs.

9.7.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.
- You [mirrored the images for a disconnected installation](#) to your registry and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You have an existing VPC in GCP. While installing a cluster in a restricted network that uses installer-provisioned infrastructure, you cannot use the installer-provisioned VPC. You must use a user-provisioned VPC that satisfies one of the following requirements:
 - Contains the mirror registry
 - Has firewall rules or a peering connection to access the mirror registry hosted elsewhere
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to. While you might need to grant access to more sites, you must grant access to ***.googleapis.com** and **accounts.google.com**.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .

9.7.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

9.7.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

9.7.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

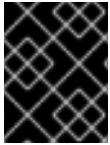
9.7.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added

to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.7.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
┌ $ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



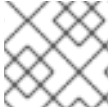
NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
 - iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
 - iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
 - v. Select the region to deploy the cluster to.
 - vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - vii. Enter a descriptive name for your cluster.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

9.7.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.7.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.22. Required parameters

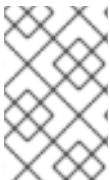
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.7.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.23. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


9.7.5.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 9.24. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
<p>credentialsMode</p>	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	<p>Mint, Passthrough, Manual or an empty string ("").</p>
<p>fips</p>	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p>	<p>false or true</p>

Parameter	Description	Values
	<p style="text-align: center;">IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p style="text-align: center;">NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

9.7.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.25. Additional GCP parameters

Parameter	Description	Values
platform.gcp.network	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.
platform.gcp.networkProjectID	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.

Parameter	Description	Values
platform.gcp.projectID	The name of the GCP project where the installation program installs the cluster.	String.
platform.gcp.region	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
platform.gcp.controlPlaneSubnet	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
platform.gcp.computeSubnet	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
platform.gcp.createFirewallRules	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
platform.gcp.publicDNSZone.project	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.
platform.gcp.publicDNSZone.id	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.

Parameter	Description	Values
<code>platform.gcp.privateDNSZone.project</code>	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the host project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the private DNS zone.
<code>platform.gcp.privateDNSZone.id</code>	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
<code>platform.gcp.licenses</code>	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: center;">  <div style="margin-left: 10px;"> <p>IMPORTANT</p> <p>The licenses parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
<code>platform.gcp.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImage.name</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.location	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID</code>	The ID of the project in which the KMS key ring exists. This value defaults to the value of the <code>platform.gcp.projectID</code> parameter if it is not set.	The GCP project ID.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyServiceAccount</code>	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name</code>	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyLocation</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.serviceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDisk.sizeGB</code>	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
<code>controlPlane.platform.gcp.osDisk.type</code>	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
<code>controlPlane.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
<code>controlPlane.platform.gcp.osImage.name</code>	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use <code>controlPlane.platform.gcp.osImage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>controlPlane.platform.gcp.tags</code>	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
<code>controlPlane.platform.gcp.type</code>	The GCP machine type for control plane machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
controlPlane.platform.gcp.zones	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
compute.platform.gcp.osDisk.encryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
compute.platform.gcp.osDisk.encryptionKey.location	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.ossDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.ossDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.ossDisk.sizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.ossDisk.type</code>	The GCP disk type for compute machines.	Either the default pd-ssd or the pd-standard disk type.
<code>compute.platform.gcp.ossImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

Parameter	Description	Values
<code>compute.platform.gcp.osimage.name</code>	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use <code>compute.platform.gcp.osimage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>compute.platform.gcp.tags</code>	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for compute machines.	One or more strings, for example <code>compute-network-tag1</code> .
<code>compute.platform.gcp.type</code>	The GCP machine type for compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example <code>n1-standard-4</code> .
<code>compute.platform.gcp.zones</code>	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .

9.7.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.26. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.7.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.23. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.7.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

As part of the installation process, you specify the custom machine type in the **install-config.yaml** file.

Sample install-config.yaml file with a custom machine type

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.7.5.5. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
    osDisk:
      diskType: pd-ssd
      diskSizeGB: 1024
```

```
    encryptionKey: 5
    kmsKey:
      name: worker-key
      keyRing: test-machine-keys
      location: global
      projectID: project-id
  tags: 6
  - control-plane-tag1
  - control-plane-tag2
  osImage: 7
    project: example-project-name
    name: example-image-name
  replicas: 3
compute: 8 9
- hyperthreading: Enabled 10
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 11
        kmsKey:
          name: worker-key
          keyRing: test-machine-keys
          location: global
          projectID: project-id
      tags: 12
      - compute-tag1
      - compute-tag2
      osImage: 13
        project: example-project-name
        name: example-image-name
    replicas: 3
metadata:
  name: test-cluster 14
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 15
  serviceNetwork:
    - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 16
    region: us-central1 17
    defaultMachinePlatform:
```



```

tags: 18
- global-tag1
- global-tag2
osImage: 19
  project: example-project-name
  name: example-image-name
network: existing_vpc 20
controlPlaneSubnet: control_plane_subnet 21
computeSubnet: compute_subnet 22
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 23
fips: false 24
sshKey: ssh-ed25519 AAAA... 25
additionalTrustBundle: | 26
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
imageContentSources: 27
- mirrors:
- <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
- <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev

```

1 14 16 17 Required. The installation program prompts you for this value.

2 8 If you do not provide these parameters and values, the installation program provides the default value.

3 9 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 10 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

5 11 Optional: The custom encryption key section to encrypt both virtual machines and persistent volumes. Your default compute service account must have the permissions granted to use your KMS key and have the correct IAM role assigned. The default service account name follows the **service-<project_number>@compute-system.iam.gserviceaccount.com** pattern. For more information about granting the correct permissions for your service account, see "Machine management" → "Creating compute machine sets" → "Creating a compute machine set on GCP".

6 12 18 Optional: A set of network tags to apply to the control plane or compute machine sets. The

- 7 13 19 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image for the installation program to use to boot control plane and compute machines. The **project** and **name**
- 15 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 20 Specify the name of an existing VPC.
- 21 Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.
- 22 Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.
- 23 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 24 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 25 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 26 Provide the contents of the certificate file that you used for your mirror registry.
- 27 Provide the **imageContentSources** section from the output of the command to mirror the repository.

9.7.5.6. Create an Ingress Controller with global access on GCP

You can create an Ingress Controller that has global access to a Google Cloud Platform (GCP) cluster. Global access is only available to Ingress Controllers using internal load balancers.

Prerequisites

- You created the **install-config.yaml** and complete any modifications to it.

Procedure

Create an Ingress Controller with global access on a new GCP cluster.

1. Change to the directory that contains the installation program and create a manifest file:

```
$ ./openshift-install create manifests --dir <installation_directory> ❶
```

- ❶ For **<installation_directory>**, specify the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a file that is named **cluster-ingress-default-ingresscontroller.yaml** in the **<installation_directory>/manifests/** directory:

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml ❶
```

- ❶ For **<installation_directory>**, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

Example output

```
cluster-ingress-default-ingresscontroller.yaml
```

3. Open the **cluster-ingress-default-ingresscontroller.yaml** file in an editor and enter a custom resource (CR) that describes the Operator configuration you want:

Sample clientAccess configuration to Global

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      providerParameters:
        gcp:
          clientAccess: Global ❶
          type: GCP
        scope: Internal ❷
      type: LoadBalancerService
```

- ❶ Set **gcp.clientAccess** to **Global**.

- ❷ Global access is only available to Ingress Controllers using internal load balancers.

9.7.5.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

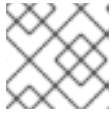
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then

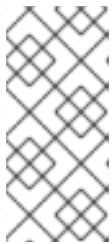
creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.7.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GCLOUD_KEYFILE_JSON** environment variables
 - The `~/gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
```

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.7.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

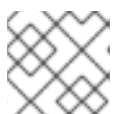
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:


```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.7.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

9.7.9. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

9.7.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.7.11. Next steps

- [Validate an installation](#).
- [Customize your cluster](#).
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, see [Registering your disconnected cluster](#)

9.8. INSTALLING A CLUSTER ON GCP INTO AN EXISTING VPC

In OpenShift Container Platform version 4.12, you can install a cluster into an existing Virtual Private Cloud (VPC) on Google Cloud Platform (GCP). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

9.8.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

9.8.2. About using a custom VPC

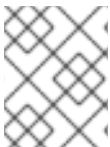
In OpenShift Container Platform 4.12, you can deploy a cluster into existing subnets in an existing Virtual Private Cloud (VPC) in Google Cloud Platform (GCP). By deploying OpenShift Container Platform into an existing GCP VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option. You must configure networking for the subnets.

9.8.2.1. Requirements for using your VPC

The union of the VPC CIDR block and the machine network CIDR must be non-empty. The subnets must be within the machine network.

The installation program does not create the following components:

- NAT gateways
- Subnets
- Route tables
- VPC network



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

9.8.2.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide one subnet for control-plane machines and one subnet for compute machines.
- The subnet's CIDRs belong to the machine CIDR that you specified.

9.8.2.3. Division of permissions

Some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

9.8.2.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed to the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

9.8.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.8.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.8.5. Obtaining the installation program

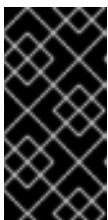
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

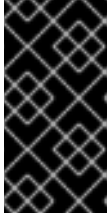
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.8.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

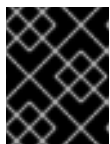
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
 - iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
 - iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
 - v. Select the region to deploy the cluster to.
 - vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - vii. Enter a descriptive name for your cluster.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.8.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.8.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.27. Required parameters

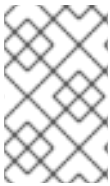
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.8.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.28. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


9.8.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 9.29. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <div data-bbox="486 656 595 1034" style="border: 1px solid gray; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div data-bbox="486 1084 595 1429" style="border: 1px solid gray; padding: 5px;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings

Parameter	Description	Values
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p>  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

9.8.6.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.30. Additional GCP parameters

Parameter	Description	Values
platform.gcp.networkork	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.
platform.gcp.networkProjectID	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.
platform.gcp.projectID	The name of the GCP project where the installation program installs the cluster.	String.

Parameter	Description	Values
<code>platform.gcp.region</code>	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
<code>platform.gcp.controlPlaneSubnet</code>	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
<code>platform.gcp.computeSubnet</code>	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
<code>platform.gcp.createFirewallRules</code>	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
<code>platform.gcp.publicDNSZone.project</code>	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.
<code>platform.gcp.publicDNSZone.id</code>	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.

Parameter	Description	Values
<code>platform.gcp.privateDNSZone.project</code>	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the <code>roles/dns.admin</code> privilege in the host project. If you do not set this value, it defaults to <code>gcp.projectId</code> .	The name of the project that contains the private DNS zone.
<code>platform.gcp.privateDNSZone.id</code>	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
<code>platform.gcp.licenses</code>	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>The <code>licenses</code> parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
<code>platform.gcp.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImage.name</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing</code>	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.location</code>	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID</code>	The ID of the project in which the KMS key ring exists. This value defaults to the value of the <code>platform.gcp.projectID</code> parameter if it is not set.	The GCP project ID.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.ServiceAccount</code>	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name</code>	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyLocation</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.serviceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
controlPlane.platform.gcp.osDiskSize GB	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
controlPlane.platform.gcp.osDiskType	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
controlPlane.platform.gcp.osImage.project	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
controlPlane.platform.gcp.osImage.name	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use controlPlane.platform.gcp.osImage.project , this field is required.	String. The name of the RHCOS image.
controlPlane.platform.gcp.tags	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.tags parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
controlPlane.platform.gcp.type	The GCP machine type for control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.type parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
controlPlane.platform.gcp.zones	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
compute.platform.gcp.osDisk.encryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyLocation	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.osDisk.sizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.osDisk.type</code>	The GCP disk type for compute machines.	Either the default pd-ssd or the pd-standard disk type.
<code>compute.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

Parameter	Description	Values
<code>compute.platform.gcp.osimage.name</code>	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use <code>compute.platform.gcp.osimage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>compute.platform.gcp.tags</code>	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for compute machines.	One or more strings, for example <code>compute-network-tag1</code> .
<code>compute.platform.gcp.type</code>	The GCP machine type for compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example <code>n1-standard-4</code> .
<code>compute.platform.gcp.zones</code>	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .

9.8.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.31. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.8.6.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.24. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.8.6.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

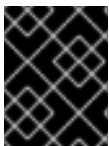
As part of the installation process, you specify the custom machine type in the **install-config.yaml** file.

Sample install-config.yaml file with a custom machine type

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.8.6.5. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
    osDisk:
      diskType: pd-ssd
      diskSizeGB: 1024
```

```
  encryptionKey: 5
    kmsKey:
      name: worker-key
      keyRing: test-machine-keys
      location: global
      projectID: project-id
  tags: 6
  - control-plane-tag1
  - control-plane-tag2
  osImage: 7
    project: example-project-name
    name: example-image-name
  replicas: 3
compute: 8 9
- hyperthreading: Enabled 10
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
    osDisk:
      diskType: pd-standard
      diskSizeGB: 128
      encryptionKey: 11
        kmsKey:
          name: worker-key
          keyRing: test-machine-keys
          location: global
          projectID: project-id
      tags: 12
      - compute-tag1
      - compute-tag2
    osImage: 13
      project: example-project-name
      name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 14
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 15
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 16
    region: us-central1 17
    defaultMachinePlatform:
```

```

tags: 18
- global-tag1
- global-tag2
osImage: 19
  project: example-project-name
  name: example-image-name
network: existing_vpc 20
controlPlaneSubnet: control_plane_subnet 21
computeSubnet: compute_subnet 22
pullSecret: '{"auths": ...}' 23
fips: false 24
sshKey: ssh-ed25519 AAAA... 25

```

1 14 16 17 23 Required. The installation program prompts you for this value.

2 8 If you do not provide these parameters and values, the installation program provides the default value.

3 9 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 10 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

5 11 Optional: The custom encryption key section to encrypt both virtual machines and persistent volumes. Your default compute service account must have the permissions granted to use your KMS key and have the correct IAM role assigned. The default service account name follows the **service-<project_number>@compute-system.iam.gserviceaccount.com** pattern. For more information about granting the correct permissions for your service account, see "Machine management" → "Creating compute machine sets" → "Creating a compute machine set on GCP".

6 12 18 Optional: A set of network tags to apply to the control plane or compute machine sets. The **platform.gcp.defaultMachinePlatform.tags** parameter will apply to both control plane and compute machines. If the **compute.platform.gcp.tags** or **controlPlane.platform.gcp.tags** parameters are set, they override the **platform.gcp.defaultMachinePlatform.tags** parameter.

7 13 19 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image for the installation program to use to boot control plane and compute machines. The **project** and **name** parameters under **platform.gcp.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the **project** and **name** parameters under **controlPlane.platform.gcp.osImage** or **compute.platform.gcp.osImage** are set, they override the **platform.gcp.defaultMachinePlatform.osImage** parameters.

- 15 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 20 Specify the name of an existing VPC.
- 21 Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.
- 22 Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.
- 24 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 25 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

Additional resources

- [Enabling customer-managed encryption keys for a compute machine set](#)

9.8.6.6. Create an Ingress Controller with global access on GCP

You can create an Ingress Controller that has global access to a Google Cloud Platform (GCP) cluster. Global access is only available to Ingress Controllers using internal load balancers.

Prerequisites

- You created the **install-config.yaml** and complete any modifications to it.

Procedure

Create an Ingress Controller with global access on a new GCP cluster.

1. Change to the directory that contains the installation program and create a manifest file:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```


- 1 For **<installation_directory>**, specify the name of the directory that contains the **install-config.yaml** file for your cluster.
2. Create a file that is named **cluster-ingress-default-ingresscontroller.yaml** in the **<installation_directory>/manifests/** directory:

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

- 1 For **<installation_directory>**, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

Example output

```
cluster-ingress-default-ingresscontroller.yaml
```

3. Open the **cluster-ingress-default-ingresscontroller.yaml** file in an editor and enter a custom resource (CR) that describes the Operator configuration you want:

Sample clientAccess configuration to Global

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      providerParameters:
        gcp:
          clientAccess: Global 1
          type: GCP
        scope: Internal 2
      type: LoadBalancerService
```

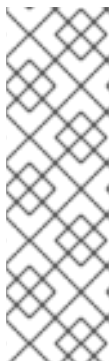
- 1 Set **gcp.clientAccess** to **Global**.
- 2 Global access is only available to Ingress Controllers using internal load balancers.

9.8.6.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

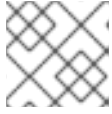
Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

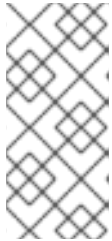
- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and

user-ca-bundle config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



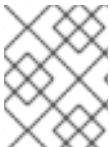
NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.8.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:

- The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GKLOUD_KEYFILE_JSON** environment variables
 - The `~/gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
```

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

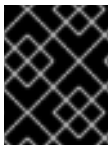


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.8.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.8.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.8.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.8.11. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

9.9. INSTALLING A CLUSTER ON GCP INTO A SHARED VPC

In OpenShift Container Platform version 4.12, you can install a cluster into a shared Virtual Private Cloud (VPC) on Google Cloud Platform (GCP). In this installation method, the cluster is configured to use a VPC from a different GCP project. A shared VPC enables an organization to connect resources from multiple projects to a common VPC network. You can communicate within the organization securely and efficiently by using internal IP addresses from that network. For more information about shared VPC, see [Shared VPC overview in the GCP documentation](#) .

The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.



IMPORTANT

Installing a cluster on GCP into a shared VPC is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

9.9.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .
- You have a GCP host project which contains a shared VPC network.
- You [configured a GCP project](#) to host the cluster. This project, known as the service project, must be attached to the host project. For more information, see [Attaching service projects in the GCP documentation](#).

- You have a GCP service account that has the [required GCP permissions](#) in the host project.

9.9.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

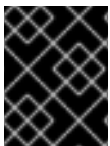
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.9.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

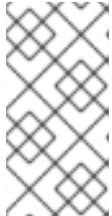
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.9.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

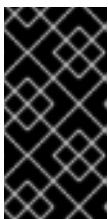
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities,

including Quay.io, which serves the container images for OpenShift Container Platform components.

9.9.5. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) into a shared VPC, you must generate the **install-config.yaml** file and modify it so that the cluster uses the correct VPC networks, DNS zones, and project names.

9.9.5.1. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

9.9.5.2. Sample customized install-config.yaml file for shared VPC installation

There are several configuration parameters which are required to install OpenShift Container Platform on GCP using a shared VPC. The following is a sample **install-config.yaml** file which demonstrates these fields.



IMPORTANT

This sample YAML file is provided for reference only. You must modify this file with the correct values for your environment and cluster.

```

apiVersion: v1
baseDomain: example.com
credentialsMode: Passthrough 1
metadata:
  name: cluster_name
platform:
  gcp:
    computeSubnet: shared-vpc-subnet-1 2
    controlPlaneSubnet: shared-vpc-subnet-2 3
    createFirewallRules: Disabled 4
    network: shared-vpc 5
    networkProjectID: host-project-name 6
    publicDNSZone:
      id: public-dns-zone 7
      project: host-project-name 8
    projectID: service-project-name 9
    region: us-east1
    defaultMachinePlatform:
      tags: 10
      - global-tag1
  controlPlane:
    name: master
    platform:
      gcp:
        tags: 11
        - control-plane-tag1
        type: n2-standard-4
        zones:
          - us-central1-a
          - us-central1-c
      replicas: 3
  compute:
    - name: worker
      platform:
        gcp:
          tags: 12
          - compute-tag1
          type: n2-standard-4
          zones:
            - us-central1-a
            - us-central1-c
          replicas: 3
  networking:
    clusterNetwork:

```

```

- cidr: 10.128.0.0/14
  hostPrefix: 23
  machineNetwork:
- cidr: 10.0.0.0/16
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA... 13

```

- 1 **credentialsMode** must be set to **Passthrough** to allow the cluster to use the provided GCP service account after cluster creation. See the "Prerequisites" section for the required GCP permissions that your service account must have.
- 2 The name of the subnet in the shared VPC for compute machines to use.
- 3 The name of the subnet in the shared VPC for control plane machines to use.
- 4 Optional. If you set **createFirewallRules** to **Disabled**, you can create and manage firewall rules manually through the use of network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have **roles/compute.networkAdmin** and **roles/compute.securityAdmin** privileges in the host project to perform these tasks automatically. If your service account does not have the **roles/dns.admin** privilege in the host project, it must have the **dns.networks.bindPrivateDNSZone** permission.
- 5 The name of the shared VPC.
- 6 The name of the host project where the shared VPC exists.
- 7 Optional. The name of a public DNS zone in the host project. If you set this value, your service account must have the **roles/dns.admin** privilege in the host project. The public DNS zone domain must match the **baseDomain** parameter. If you do not set this value, the installation program will use the public DNS zone in the service project.
- 8 Optional. The name of the host project which contains the public DNS zone. This value is required if you specify a public DNS zone that exists in another project.
- 9 The name of the GCP project where you want to install the cluster.
- 10 11 12 Optional. If you want to manually create and manage your GCP firewall rules, you can set **platform.gcp.createFirewallRules** to **Disabled** and then specify one or more network tags. You can set tags on the compute machines, the control plane machines, or all machines.
- 13 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

9.9.5.3. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.9.5.3.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.32. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>. <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.9.5.3.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.33. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


9.9.5.3.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 9.34. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1440 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...


9.9.5.3.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.35. Additional GCP parameters

Parameter	Description	Values
platform.gcp.network	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.

Parameter	Description	Values
<code>platform.gcp.networkProjectID</code>	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.
<code>platform.gcp.projectID</code>	The name of the GCP project where the installation program installs the cluster.	String.
<code>platform.gcp.region</code>	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
<code>platform.gcp.controlPlaneSubnet</code>	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
<code>platform.gcp.computeSubnet</code>	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
<code>platform.gcp.createFirewallRules</code>	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
<code>platform.gcp.publicDNSZone.project</code>	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.

Parameter	Description	Values
<code>platform.gcp.publicDNSZoneId</code>	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.
<code>platform.gcp.privateDNSZoneProject</code>	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the host project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the private DNS zone.
<code>platform.gcp.privateDNSZoneId</code>	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
<code>platform.gcp.licenses</code>	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The licenses parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
<code>platform.gcp.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImageProject</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImageName</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImageProject</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing</code>	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.location</code>	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID</code>	The ID of the project in which the KMS key ring exists. This value defaults to the value of the <code>platform.gcp.projectID</code> parameter if it is not set.	The GCP project ID.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyServiceAccount</code>	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name</code>	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyLocation</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.serviceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
controlPlane.platform.gcp.osDiskSize GB	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
controlPlane.platform.gcp.osDiskType	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
controlPlane.platform.gcp.osImage.project	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
controlPlane.platform.gcp.osImage.name	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use controlPlane.platform.gcp.osImage.project , this field is required.	String. The name of the RHCOS image.
controlPlane.platform.gcp.tags	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.tags parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
controlPlane.platform.gcp.type	The GCP machine type for control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.type parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
<code>controlPlane.platform.gcp.zones</code>	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing</code>	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKeyLocation</code>	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.osDisk.diskSizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.osDisk.diskType</code>	The GCP disk type for compute machines.	Either the default pd-ssd or the pd-standard disk type.
<code>compute.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

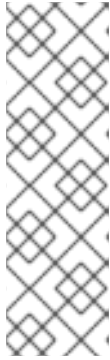
Parameter	Description	Values
<code>compute.platform.gcp.osimage.name</code>	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use <code>compute.platform.gcp.osimage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>compute.platform.gcp.tags</code>	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for compute machines.	One or more strings, for example <code>compute-network-tag1</code> .
<code>compute.platform.gcp.type</code>	The GCP machine type for compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example <code>n1-standard-4</code> .
<code>compute.platform.gcp.zones</code>	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .

9.9.5.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object's `spec.noProxy` field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.9.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GCPLOUD_KEYFILE_JSON** environment variables
 - The **~/gcp/osServiceAccount.json** file
 - The **gcloud cli** default credentials
2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

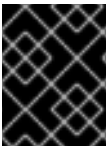


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.9.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.9.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

9.9.9. Optional: Adding Ingress DNS records for shared VPC installations

If the public DNS zone exists in a host project outside the project where you installed your cluster, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You completed the installation of OpenShift Container Platform on GCP into a shared VPC.
- Your public DNS zone exists in a host project separate from the service project that contains your cluster.

Procedure

1. Verify that the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field by running the following command:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
router-default LoadBalancer  172.30.18.154  35.233.157.184 80:32288/TCP,443:31215/TCP 98
```

- Record the external IP address of the router by running the following command:

```
$ oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'
```

- Add a record to your GCP public zone with the router's external IP address and the name ***.apps.<cluster_name>.<cluster_domain>**. You can use the **gcloud** command line utility or the GCP web console.
- To add manual records instead of a wildcard record, create entries for each of the cluster's current routes. You can gather these routes by running the following command:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.9.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.9.11. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

9.10. INSTALLING A PRIVATE CLUSTER ON GCP

In OpenShift Container Platform version 4.12, you can install a private cluster into an existing VPC on Google Cloud Platform (GCP). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

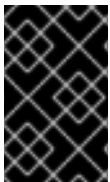
9.10.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured a GCP project](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .

9.10.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
 - The API services for the cloud to which you provision.
 - The hosts on the network that you provision.

- The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

9.10.2.1. Private clusters in GCP

To create a private cluster on Google Cloud Platform (GCP), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

The cluster still requires access to internet to access the GCP APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public network load balancers, which support public ingress
- A public DNS zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private DNS zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

Because it is not possible to limit access to external load balancers based on source tags, the private cluster uses only internal load balancers to allow access to internal instances.

The internal load balancer relies on instance groups rather than the target pools that the network load balancers use. The installation program creates instance groups for each zone, even if there is no instance in that group.

- The cluster IP address is internal only.
- One forwarding rule manages both the Kubernetes API and machine config server ports.
- The backend service is comprised of each zone's instance group and, while it exists, the bootstrap instance group.
- The firewall uses a single rule that is based on only internal source ranges.

9.10.2.1.1. Limitations

No health check for the Machine config server, **/healthz**, runs because of a difference in load balancer functionality. Two internal load balancers cannot share a single IP address, but two network load balancers can share a single external IP address. Instead, the health of an instance is determined entirely by the **/readyz** check on port 6443.

9.10.3. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into an existing VPC in Google Cloud Platform (GCP). If you do, you must also use existing subnets within the VPC and routing rules.

By deploying OpenShift Container Platform into an existing GCP VPC, you might be able to avoid limit

constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself.

9.10.3.1. Requirements for using your VPC

The installation program will no longer create the following components:

- VPC
- Subnets
- Cloud router
- Cloud NAT
- NAT IP addresses

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VPC options like DHCP, so you must do so before you install the cluster.

Your VPC and subnets must meet the following characteristics:

- The VPC must be in the same GCP project that you deploy the OpenShift Container Platform cluster to.
- To allow access to the internet from the control plane and compute machines, you must configure cloud NAT on the subnets to allow egress to it. These machines do not have a public address. Even if you do not require access to the internet, you must allow egress to the VPC network to obtain the installation program and images. Because multiple cloud NATs cannot be configured on the shared subnets, the installation program cannot configure it.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist and belong to the VPC that you specified.
- The subnet CIDRs belong to the machine CIDR.
- You must provide a subnet to deploy the cluster control plane and compute machines to. You can use the same subnet for both machine types.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted.

9.10.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or Ingress rules.

The GCP credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets,

routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage, and nodes.

9.10.3.3. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is preserved by firewall rules that reference the machines in your cluster by the cluster's infrastructure ID. Only traffic within the cluster is allowed.

If you deploy multiple clusters to the same VPC, the following components might share access between clusters:

- The API, which is globally available with an external publishing strategy or available throughout the network in an internal publishing strategy
- Debugging tools, such as ports on VM instances that are open to the machine CIDR for SSH and ICMP access

9.10.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

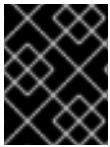
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.10.5. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

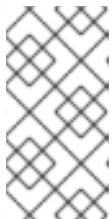
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

■

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.10.6. Obtaining the installation program

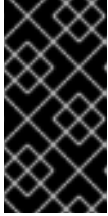
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

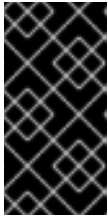
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.10.7. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

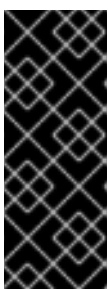
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

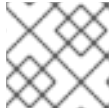
1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

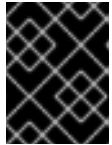
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

9.10.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

9.10.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 9.36. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

9.10.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 9.37. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


9.10.7.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 9.38. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
<p>credentialsMode</p>	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. If you are installing on GCP into a shared virtual private cloud (VPC), credentialsMode must be set to Passthrough.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	<p>Mint, Passthrough, Manual or an empty string ("").</p>

Parameter	Description	Values
<p>fips</p>	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	<p>false or true</p>

Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

9.10.7.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 9.39. Additional GCP parameters

Parameter	Description	Values
platform.gcp.networkork	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set platform.gcp.networkProjectID with the name of the GCP project that contains the shared VPC.	String.

Parameter	Description	Values
<code>platform.gcp.networkProjectID</code>	Optional. The name of the GCP project that contains the shared VPC where you want to deploy your cluster.	String.
<code>platform.gcp.projectID</code>	The name of the GCP project where the installation program installs the cluster.	String.
<code>platform.gcp.region</code>	The name of the GCP region that hosts your cluster.	Any valid region name, such as us-central1 .
<code>platform.gcp.controlPlaneSubnet</code>	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
<code>platform.gcp.computeSubnet</code>	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.
<code>platform.gcp.createFirewallRules</code>	Optional. Set this value to Disabled if you want to create and manage your firewall rules using network tags. By default, the cluster will automatically create and manage the firewall rules that are required for cluster communication. Your service account must have roles/compute.networkAdmin and roles/compute.securityAdmin privileges in the host project to perform these tasks automatically. If your service account does not have the roles/dns.admin privilege in the host project, it must have the dns.networks.bindPrivateDNSZone permission.	Enabled or Disabled . The default value is Enabled .
<code>platform.gcp.publicDNSZone.project</code>	Optional. The name of the project that contains the public DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the specified project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the public DNS zone.

Parameter	Description	Values
<code>platform.gcp.publicDNSZoneId</code>	Optional. The ID or name of an existing public DNS zone. The public DNS zone domain must match the baseDomain parameter. If you do not set this value, the installation program will use a public DNS zone in the service project.	The public DNS zone name.
<code>platform.gcp.privateDNSZone.project</code>	Optional. The name of the project that contains the private DNS zone. If you set this value, your service account must have the roles/dns.admin privilege in the host project. If you do not set this value, it defaults to gcp.projectId .	The name of the project that contains the private DNS zone.
<code>platform.gcp.privateDNSZoneId</code>	Optional. The ID or name of an existing private DNS zone. If you do not set this value, the installation program will create a private DNS zone in the service project.	The private DNS zone name.
<code>platform.gcp.licenses</code>	<p>A list of license URLs that must be applied to the compute images.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>The licenses parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.</p> </div> </div>	Any license available with the license API , such as the license to enable nested virtualization . You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installation program to copy the source image before use.
<code>platform.gcp.defaultMachinePlatform.zones</code>	The availability zones where the installation program creates machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskSizeGB</code>	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.
<code>platform.gcp.defaultMachinePlatform.osDiskType</code>	The GCP disk type .	Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. Compute nodes can be either type.
<code>platform.gcp.defaultMachinePlatform.osImageProject</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for both types of machines.	String. The name of GCP project where the image is located.
<code>platform.gcp.defaultMachinePlatform.osImageName</code>	The name of the custom RHCOS image for the installation program to use to boot control plane and compute machines. If you use <code>platform.gcp.defaultMachinePlatform.osImageProject</code> , this field is required.	String. The name of the RHCOS image.
<code>platform.gcp.defaultMachinePlatform.tags</code>	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example network-tag1 .

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.type</code>	The GCP machine type for control plane and compute machines.	The GCP machine type, for example n1-standard-4 .
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyName</code>	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyRing</code>	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKeyLocation</code>	The GCP location in which the KMS key ring exists.	The GCP location.

Parameter	Description	Values
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.projectID</code>	The ID of the project in which the KMS key ring exists. This value defaults to the value of the <code>platform.gcp.projectID</code> parameter if it is not set.	The GCP project ID.
<code>platform.gcp.defaultMachinePlatform.osDiskEncryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>controlPlane.platform.gcp.osDiskEncryptionKey.kmsKey.name</code>	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyRing</code>	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.kmsKeyLocation</code>	For control plane machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.projectID</code>	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>controlPlane.platform.gcp.osDisk.encryptionKey.serviceAccount</code>	The GCP service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .

Parameter	Description	Values
controlPlane.platform.gcp.osDiskSize GB	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
controlPlane.platform.gcp.osDiskType	The GCP disk type for control plane machines.	Control plane machines must use the pd-ssd disk type, which is the default.
controlPlane.platform.gcp.osImage.project	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for control plane machines only.	String. The name of GCP project where the image is located.
controlPlane.platform.gcp.osImage.name	The name of the custom RHCOS image for the installation program to use to boot control plane machines. If you use controlPlane.platform.gcp.osImage.project , this field is required.	String. The name of the RHCOS image.
controlPlane.platform.gcp.tags	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.tags parameter for control plane machines.	One or more strings, for example control-plane-tag1 .
controlPlane.platform.gcp.type	The GCP machine type for control plane machines. If set, this parameter overrides the platform.gcp.defaultMachinePlatform.type parameter.	The GCP machine type, for example n1-standard-4 .

Parameter	Description	Values
controlPlane.platform.gcp.zones	The availability zones where the installation program creates control plane machines.	A list of valid GCP availability zones , such as us-central1-a , in a YAML sequence .
compute.platform.gcp.osDisk.encryptionKey.kmsKeyName	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyRing	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.
compute.platform.gcp.osDisk.encryptionKey.kmsKeyLocation	For compute machines, the GCP location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations .	The GCP location for the key ring.

Parameter	Description	Values
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.projectID</code>	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The GCP project ID.
<code>compute.platform.gcp.osDisk.encryptionKey.kmsKey.serviceAccount</code>	The GCP service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about GCP service accounts, see Google's documentation on service accounts .	The GCP service account email, for example <code><service_account_name>@<project_id>.iam.gserviceaccount.com</code> .
<code>compute.platform.gcp.osDisk.diskSizeGB</code>	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.
<code>compute.platform.gcp.osDisk.diskType</code>	The GCP disk type for compute machines.	Either the default <code>pd-ssd</code> or the <code>pd-standard</code> disk type.
<code>compute.platform.gcp.osImage.project</code>	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image for the installation program to use for compute machines only.	String. The name of GCP project where the image is located.

Parameter	Description	Values
<code>compute.platform.gcp.osimage.name</code>	The name of the custom RHCOS image for the installation program to use to boot compute machines. If you use <code>compute.platform.gcp.osimage.project</code> , this field is required.	String. The name of the RHCOS image.
<code>compute.platform.gcp.tags</code>	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.tags</code> parameter for compute machines.	One or more strings, for example <code>compute-network-tag1</code> .
<code>compute.platform.gcp.type</code>	The GCP machine type for compute machines. If set, this parameter overrides the <code>platform.gcp.defaultMachinePlatform.type</code> parameter.	The GCP machine type, for example <code>n1-standard-4</code> .
<code>compute.platform.gcp.zones</code>	The availability zones where the installation program creates compute machines.	A list of valid GCP availability zones , such as <code>us-central1-a</code> , in a YAML sequence .

9.10.7.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.40. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

[1] For the control plane, the number of vCPUs must be at least 4. For the compute nodes, the number of vCPUs must be at least 2.

[2] For the control plane, the number of IOPS must be at least 300. For the compute nodes, the number of IOPS must be at least 300.

[3] For RHEL 8.6 and later, the number of vCPUs must be at least 2. For RHEL 8.5 and earlier, the number of vCPUs must be at least 4.

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.10.7.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.25. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.10.7.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

As part of the installation process, you specify the custom machine type in the **install-config.yaml** file.

Sample install-config.yaml file with a custom machine type

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.10.7.5. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
```

```
- us-central1-c
osDisk:
  diskType: pd-ssd
  diskSizeGB: 1024
  encryptionKey: 5
  kmsKey:
    name: worker-key
    keyRing: test-machine-keys
    location: global
    projectID: project-id
  tags: 6
- control-plane-tag1
- control-plane-tag2
osImage: 7
  project: example-project-name
  name: example-image-name
replicas: 3
compute: 8 9
- hyperthreading: Enabled 10
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 11
        kmsKey:
          name: worker-key
          keyRing: test-machine-keys
          location: global
          projectID: project-id
      tags: 12
        - compute-tag1
        - compute-tag2
      osImage: 13
        project: example-project-name
        name: example-image-name
      replicas: 3
  metadata:
    name: test-cluster 14
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 15
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    gcp:
```

```

projectID: openshift-production 16
region: us-central1 17
defaultMachinePlatform:
  tags: 18
  - global-tag1
  - global-tag2
  osImage: 19
    project: example-project-name
    name: example-image-name
network: existing_vpc 20
controlPlaneSubnet: control_plane_subnet 21
computeSubnet: compute_subnet 22
pullSecret: '{"auths": ...}' 23
fips: false 24
sshKey: ssh-ed25519 AAAA... 25
publish: Internal 26

```

- 1 14 16 17 23 Required. The installation program prompts you for this value.
- 2 8 If you do not provide these parameters and values, the installation program provides the default value.
- 3 9 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 4 10 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

- 5 11 Optional: The custom encryption key section to encrypt both virtual machines and persistent volumes. Your default compute service account must have the permissions granted to use your KMS key and have the correct IAM role assigned. The default service account name follows the **service-<project_number>@compute-system.iam.gserviceaccount.com** pattern. For more information about granting the correct permissions for your service account, see "Machine management" → "Creating compute machine sets" → "Creating a compute machine set on GCP".
- 6 12 18 Optional: A set of network tags to apply to the control plane or compute machine sets. The **platform.gcp.defaultMachinePlatform.tags** parameter will apply to both control plane and compute machines. If the **compute.platform.gcp.tags** or **controlPlane.platform.gcp.tags** parameters are set, they override the **platform.gcp.defaultMachinePlatform.tags** parameter.
- 7 13 19 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image for the installation program to use to boot control plane and compute machines. The **project** and **name** parameters under **platform.gcp.defaultMachinePlatform.osImage** apply to both control plane

and compute machines. If the **project** and **name** parameters under **controlPlane.platform.gcp.osImage** or **compute.platform.gcp.osImage** are set, they override the **platform.gcp.defaultMachinePlatform.osImage** parameters.

- 15 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 20 Specify the name of an existing VPC.
- 21 Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.
- 22 Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.
- 24 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Progress cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 25 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 26 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

Additional resources

- [Enabling customer-managed encryption keys for a compute machine set](#)

9.10.7.6. Create an Ingress Controller with global access on GCP

You can create an Ingress Controller that has global access to a Google Cloud Platform (GCP) cluster. Global access is only available to Ingress Controllers using internal load balancers.

Prerequisites

- You created the **install-config.yaml** and complete any modifications to it.

Procedure

Create an Ingress Controller with global access on a new GCP cluster.

1. Change to the directory that contains the installation program and create a manifest file:

```
$ ./openshift-install create manifests --dir <installation_directory> ❶
```

- ❶ For **<installation_directory>**, specify the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a file that is named **cluster-ingress-default-ingresscontroller.yaml** in the **<installation_directory>/manifests/** directory:

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml ❶
```

- ❶ For **<installation_directory>**, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

Example output

```
cluster-ingress-default-ingresscontroller.yaml
```

3. Open the **cluster-ingress-default-ingresscontroller.yaml** file in an editor and enter a custom resource (CR) that describes the Operator configuration you want:

Sample clientAccess configuration to Global

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      providerParameters:
        gcp:
          clientAccess: Global ❶
          type: GCP
        scope: Internal ❷
      type: LoadBalancerService
```

- ❶ Set **gcp.clientAccess** to **Global**.

- ❷ Global access is only available to Ingress Controllers using internal load balancers.

9.10.7.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then

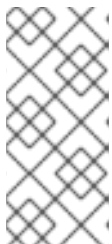
creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

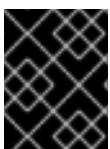


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.10.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
 - The **GOOGLE_CREDENTIALS**, **GOOGLE_CLOUD_KEYFILE_JSON**, or **GKLOUD_KEYFILE_JSON** environment variables
 - The `~/gcp/osServiceAccount.json` file
 - The **gcloud cli** default credentials
- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

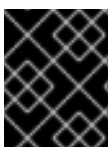
If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

- Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
 - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
 - If you included the **Service Account Key Admin** role, you can remove it.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
```

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

9.10.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.10.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.10.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct,

either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.10.12. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

9.11. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN GCP BY USING DEPLOYMENT MANAGER TEMPLATES

In OpenShift Container Platform version 4.12, you can install a cluster on Google Cloud Platform (GCP) that uses infrastructure that you provide.

The steps for performing a user-provided infrastructure install are outlined here. Several [Deployment Manager](#) templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

9.11.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#) .



NOTE

Be sure to also review this site list if you are configuring a proxy.

9.11.2. Certificate signing requests management

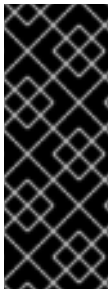
Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

9.11.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.11.4. Configuring your GCP project

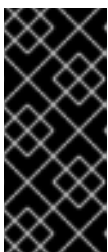
Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

9.11.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See [Creating and Managing Projects](#) in the GCP documentation.



IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the **api-int.<cluster_name>.<base_domain>** URL; the Premium Tier is required for internal load balancing.

9.11.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. You may also enable optional API services which are not required for installation. See [Enabling services](#) in the GCP documentation.

Table 9.41. Required API services

API service	Console service name
Compute Engine API	compute.googleapis.com
Cloud Resource Manager API	cloudresourcemanager.googleapis.com
Google DNS API	dns.googleapis.com
IAM Service Account Credentials API	iamcredentials.googleapis.com
Identity and Access Management (IAM) API	iam.googleapis.com
Service Usage API	serviceusage.googleapis.com

Table 9.42. Optional API services

API service	Console service name
Cloud Deployment Manager V2 API	deploymentmanager.googleapis.com
Google Cloud APIs	cloudapis.googleapis.com
Service Management API	servicemanagement.googleapis.com
Google Cloud Storage JSON API	storage-api.googleapis.com
Cloud Storage	storage-component.googleapis.com

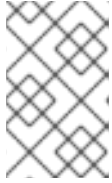
9.11.4.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform

cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.



NOTE

If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](#).

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](#) in the GCP documentation. Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](#) in the GCP documentation. You typically have four name servers.
4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](#).
5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](#) in the GCP documentation.
6. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain. This process might include a request to your company's IT department or the division that controls the root domain and DNS services for your company.

9.11.4.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default [Quotas](#) do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

Table 9.43. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	6	1
Firewall rules	Networking	Global	11	1

Service	Component	Location	Total resources required	Resources removed after bootstrap
Forwarding rules	Compute	Global	2	0
Health checks	Compute	Global	2	0
Images	Compute	Global	1	0
Networks	Networking	Global	1	0
Routers	Networking	Global	1	0
Routes	Networking	Global	2	0
Subnetworks	Compute	Global	2	0
Target pools	Networking	Global	2	0



NOTE

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europa-north1**
- **europa-west2**
- **europa-west3**
- **europa-west6**
- **northamerica-northeast1**
- **southamerica-east1**

- **us-west2**

You can increase resource quotas from the [GCP console](#), but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

9.11.4.5. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See [Creating a service account](#) in the GCP documentation.
2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See [Granting roles to a service account for specific resources](#).



NOTE

While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. You can create the service account key in JSON format, or attach the service account to a GCP virtual machine. See [Creating service account keys](#) and [Creating and enabling service accounts for instances](#) in the GCP documentation.

You must have a service account key or a virtual machine with an attached service account to create the cluster.

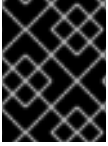


NOTE

If you use a virtual machine with an attached service account to create your cluster, you must set **credentialsMode: Manual** in the **install-config.yaml** file before installation.

9.11.4.6. Required GCP roles

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. If the security policies for your organization require a more restrictive set of permissions, you can create a service account with the following permissions.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions.

If you deploy your cluster into an existing virtual private cloud (VPC), the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- IAM Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Required roles for using passthrough credentials mode

- Compute Load Balancer Admin
- IAM Role Viewer

Required roles for user-provisioned GCP infrastructure

- Deployment Manager Editor

The roles are applied to the service accounts that the control plane and compute machines use:

Table 9.44. GCP service account permissions

Account	Roles
Control Plane	roles/compute.instanceAdmin
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	roles/compute.viewer

Account	Roles
	roles/storage.admin

9.11.4.7. Required GCP permissions for user-provisioned infrastructure

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform.

If the security policies for your organization require a more restrictive set of permissions, you can create [custom roles](#) with the necessary permissions. The following permissions are required for the user-provisioned infrastructure for creating and deleting the OpenShift Container Platform cluster.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions. For more information, see "Required roles for using passthrough credentials mode" in the "Required GCP roles" section.

Example 9.26. Required permissions for creating network resources

- **compute.addresses.create**
- **compute.addresses.createInternal**
- **compute.addresses.delete**
- **compute.addresses.get**
- **compute.addresses.list**
- **compute.addresses.use**
- **compute.addresses.useInternal**
- **compute.firewalls.create**
- **compute.firewalls.delete**
- **compute.firewalls.get**
- **compute.firewalls.list**
- **compute.forwardingRules.create**
- **compute.forwardingRules.get**
- **compute.forwardingRules.list**
- **compute.forwardingRules.setLabels**
- **compute.networks.create**
- **compute.networks.get**

- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.create**
- **compute.routers.get**
- **compute.routers.list**
- **compute.routers.update**
- **compute.routes.list**
- **compute.subnetworks.create**
- **compute.subnetworks.get**
- **compute.subnetworks.list**
- **compute.subnetworks.use**
- **compute.subnetworks.useExternallp**

Example 9.27. Required permissions for creating load balancer resources

- **compute.regionBackendServices.create**
- **compute.regionBackendServices.get**
- **compute.regionBackendServices.list**
- **compute.regionBackendServices.update**
- **compute.regionBackendServices.use**
- **compute.targetPools.addInstance**
- **compute.targetPools.create**
- **compute.targetPools.get**
- **compute.targetPools.list**
- **compute.targetPools.removeInstance**
- **compute.targetPools.use**

Example 9.28. Required permissions for creating DNS resources

- **dns.changes.create**
- **dns.changes.get**
- **dns.managedZones.create**

- `dns.managedZones.get`
- `dns.managedZones.list`
- `dns.networks.bindPrivateDNSZone`
- `dns.resourceRecordSets.create`
- `dns.resourceRecordSets.list`
- `dns.resourceRecordSets.update`

Example 9.29. Required permissions for creating Service Account resources

- `iam.serviceAccountKeys.create`
- `iam.serviceAccountKeys.delete`
- `iam.serviceAccountKeys.get`
- `iam.serviceAccountKeys.list`
- `iam.serviceAccounts.actAs`
- `iam.serviceAccounts.create`
- `iam.serviceAccounts.delete`
- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.get`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.30. Required permissions for creating compute resources

- `compute.disks.create`
- `compute.disks.get`
- `compute.disks.list`
- `compute.instanceGroups.create`
- `compute.instanceGroups.delete`
- `compute.instanceGroups.get`
- `compute.instanceGroups.list`
- `compute.instanceGroups.update`

- **compute.instanceGroups.use**
- **compute.instances.create**
- **compute.instances.delete**
- **compute.instances.get**
- **compute.instances.list**
- **compute.instances.setLabels**
- **compute.instances.setMetadata**
- **compute.instances.setServiceAccount**
- **compute.instances.setTags**
- **compute.instances.use**
- **compute.machineTypes.get**
- **compute.machineTypes.list**

Example 9.31. Required for creating storage resources

- **storage.buckets.create**
- **storage.buckets.delete**
- **storage.buckets.get**
- **storage.buckets.list**
- **storage.objects.create**
- **storage.objects.delete**
- **storage.objects.get**
- **storage.objects.list**

Example 9.32. Required permissions for creating health check resources

- **compute.healthChecks.create**
- **compute.healthChecks.get**
- **compute.healthChecks.list**
- **compute.healthChecks.useReadOnly**
- **compute.httpHealthChecks.create**
- **compute.httpHealthChecks.get**

- `compute.httpHealthChecks.list`
- `compute.httpHealthChecks.useReadOnly`

Example 9.33. Required permissions to get GCP zone and region related information

- `compute.globalOperations.get`
- `compute.regionOperations.get`
- `compute.regions.list`
- `compute.zoneOperations.get`
- `compute.zones.get`
- `compute.zones.list`

Example 9.34. Required permissions for checking services and quotas

- `monitoring.timeSeries.list`
- `serviceusage.quotas.get`
- `serviceusage.services.list`

Example 9.35. Required IAM permissions for installation

- `iam.roles.get`

Example 9.36. Required Images permissions for installation

- `compute.images.create`
- `compute.images.delete`
- `compute.images.get`
- `compute.images.list`

Example 9.37. Optional permission for running gather bootstrap

- `compute.instances.getSerialPortOutput`

Example 9.38. Required permissions for deleting network resources

- `compute.addresses.delete`
- `compute.addresses.deleteInternal`

- `compute.addresses.list`
- `compute.firewalls.delete`
- `compute.firewalls.list`
- `compute.forwardingRules.delete`
- `compute.forwardingRules.list`
- `compute.networks.delete`
- `compute.networks.list`
- `compute.networks.updatePolicy`
- `compute.routers.delete`
- `compute.routers.list`
- `compute.routes.list`
- `compute.subnetworks.delete`
- `compute.subnetworks.list`

Example 9.39. Required permissions for deleting load balancer resources

- `compute.regionBackendServices.delete`
- `compute.regionBackendServices.list`
- `compute.targetPools.delete`
- `compute.targetPools.list`

Example 9.40. Required permissions for deleting DNS resources

- `dns.changes.create`
- `dns.managedZones.delete`
- `dns.managedZones.get`
- `dns.managedZones.list`
- `dns.resourceRecordSets.delete`
- `dns.resourceRecordSets.list`

Example 9.41. Required permissions for deleting Service Account resources

- `iam.serviceAccounts.delete`

- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.42. Required permissions for deleting compute resources

- `compute.disks.delete`
- `compute.disks.list`
- `compute.instanceGroups.delete`
- `compute.instanceGroups.list`
- `compute.instances.delete`
- `compute.instances.list`
- `compute.instances.stop`
- `compute.machineTypes.list`

Example 9.43. Required for deleting storage resources

- `storage.buckets.delete`
- `storage.buckets.getIamPolicy`
- `storage.buckets.list`
- `storage.objects.delete`
- `storage.objects.list`

Example 9.44. Required permissions for deleting health check resources

- `compute.healthChecks.delete`
- `compute.healthChecks.list`
- `compute.httpHealthChecks.delete`
- `compute.httpHealthChecks.list`

Example 9.45. Required Images permissions for deletion

- `compute.images.delete`

- **compute.images.list**

Example 9.46. Required permissions to get Region related information

- **compute.regions.get**

Example 9.47. Required Deployment Manager permissions

- **deploymentmanager.deployments.create**
- **deploymentmanager.deployments.delete**
- **deploymentmanager.deployments.get**
- **deploymentmanager.deployments.list**
- **deploymentmanager.manifests.get**
- **deploymentmanager.operations.get**
- **deploymentmanager.resources.list**

Additional resources

- [Optimizing storage](#)

9.11.4.8. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **australia-southeast1** (Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)

- **europa-central2** (Warsaw, Poland)
- **europa-north1** (Hamina, Finland)
- **europa-southwest1** (Madrid, Spain)
- **europa-west1** (St. Ghislain, Belgium)
- **europa-west2** (London, England, UK)
- **europa-west3** (Frankfurt, Germany)
- **europa-west4** (Eemshaven, Netherlands)
- **europa-west6** (Zürich, Switzerland)
- **europa-west8** (Milan, Italy)
- **europa-west9** (Paris, France)
- **europa-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)



NOTE

To determine which machine type instances are available by region and zone, see the Google [documentation](#).

9.11.4.9. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

Prerequisites

- You created a project to host your cluster.
- You created a service account and granted it the required permissions.

Procedure

1. Install the following binaries in **\$PATH**:

- **gcloud**
- **gsutil**

See [Install the latest Cloud SDK version](#) in the GCP documentation.

2. Authenticate using the **gcloud** tool with your configured service account.
See [Authorizing with a service account](#) in the GCP documentation.

9.11.5. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

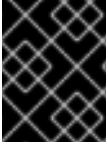
This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

9.11.5.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 9.45. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

9.11.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.46. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.11.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.48. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.11.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-`<number_of_cpus>-<amount_of_memory_in_mb>`

For example, **custom-6-20480**.

9.11.6. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

9.11.6.1. Optional: Creating a separate `/var` partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

9.11.6.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
 - iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.
 - iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
 - v. Select the region to deploy the cluster to.
 - vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - vii. Enter a descriptive name for your cluster.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
- c. Optional: If you do not want the cluster to provision compute machines, empty the compute pool by editing the resulting **install-config.yaml** file to set **replicas** to **0** for the **compute** pool:

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

1 Set to **0**.

2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.11.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxvonly** and

user-ca bundle config map in the namespace. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.11.6.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Optional: If you do not want the cluster to provision compute machines, remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yaml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

- a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yaml** file.
- b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
- c. Save and exit the file.

5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yaml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
```

```
publicZone: 2
  id: example.openshift.com
  status: {}
```

- 1 2 Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

Additional resources

- [Optional: Adding the ingress DNS records](#)

9.11.7. Exporting common variables

9.11.7.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

9.11.7.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).



NOTE

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Generate the Ignition config files for your cluster.
- Install the **jq** package.

Procedure

1. Export the following common variables to be used by the provided Deployment Manager templates:

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/17'
$ export WORKER_SUBNET_CIDR='10.0.128.0/17'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

9.11.8. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the **Deployment Manager template for the VPC** section of this topic and save it as **01_vpc.py** on your computer. This template describes the VPC that your cluster requires.
2. Create a **01_vpc.yaml** resource definition file:

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **region** is the region to deploy the cluster into, for example **us-central1**.
- 3 **master_subnet_cidr** is the CIDR for the master subnet, for example **10.0.0.0/17**.
- 4 **worker_subnet_cidr** is the CIDR for the worker subnet, for example **10.0.128.0/17**.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

9.11.8.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

Example 9.49. 01_vpc.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '${ref.' + context.properties['infra_id'] + '-master-subnet.selfLink}',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }, {
        'name': context.properties['infra_id'] + '-nat-worker',
        'natIpAllocateOption': 'AUTO_ONLY',
        'minPortsPerVm': 512,
        'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
        'subnetworks': [{
```

```

        'name': '${ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
        'sourceIpRangesToNat': ['ALL_IP_RANGES']
    }
}
}
}
return {'resources': resources}

```

9.11.9. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

9.11.9.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

9.11.9.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 9.47. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .

Protocol	Port	Description
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 9.48. Ports used for all-machine to control plane communications

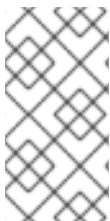
Protocol	Port	Description
TCP	6443	Kubernetes API

Table 9.49. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

9.11.10. Creating load balancers in GCP

You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the internal load balancer** section of this topic and save it as **02_lb_int.py** on your computer. This template describes the internal load balancing objects that your cluster requires.
2. For an external cluster, also copy the template from the **Deployment Manager template for the external load balancer** section of this topic and save it as **02_lb_ext.py** on your computer. This template describes the external load balancing objects that your cluster requires.
3. Export the variables that the deployment template uses:

- a. Export the cluster network location:

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
```

- b. Export the control plane subnet location:

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

- c. Export the three zones that the cluster uses:

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9`)
```

4. Create a **02_infra.yaml** resource definition file:

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py 1
```



```

resources:
- name: cluster-lb-ext 2
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' 3
    region: '${REGION}' 4
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' 5
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: 6
      - '${ZONE_0}'
      - '${ZONE_1}'
      - '${ZONE_2}'
EOF

```

1 **2** Required only when deploying an external cluster.

3 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

4 **region** is the region to deploy the cluster into, for example **us-central1**.

5 **control_subnet** is the URI to the control subnet.

6 **zones** are the zones to deploy the control plane instances into, like **us-east1-b**, **us-east1-c**, and **us-east1-d**.

5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:

```
$ export CLUSTER_IP=$(gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address)
```

7. For an external cluster, also export the cluster public IP address:

```
$ export CLUSTER_PUBLIC_IP=$(gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address)
```

9.11.10.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

Example 9.50. **02_lb_ext.py** Deployment Manager template

```
def GenerateConfig(context):
```

```

resources = [{
    'name': context.properties['infra_id'] + '-cluster-public-ip',
    'type': 'compute.v1.address',
    'properties': {
        'region': context.properties['region']
    }
}, {
    # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
    # probe for kube-apiserver
    'name': context.properties['infra_id'] + '-api-http-health-check',
    'type': 'compute.v1.httpHealthCheck',
    'properties': {
        'port': 6080,
        'requestPath': '/readyz'
    }
}, {
    'name': context.properties['infra_id'] + '-api-target-pool',
    'type': 'compute.v1.targetPool',
    'properties': {
        'region': context.properties['region'],
        'healthChecks': ['$$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
        'instances': []
    }
}, {
    'name': context.properties['infra_id'] + '-api-forwarding-rule',
    'type': 'compute.v1.forwardingRule',
    'properties': {
        'region': context.properties['region'],
        'IPAddress': '$$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
        'target': '$$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
        'portRange': '6443'
    }
}
]]

return {'resources': resources}

```

9.11.10.2. Deployment Manager template for the internal load balancer

You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

Example 9.51. 02_lb_int.py Deployment Manager template

```

def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',

```

```

    'properties': {
      'addressType': 'INTERNAL',
      'region': context.properties['region'],
      'subnetwork': context.properties['control_subnet']
    }
  }, {
    # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
    # probe for kube-apiserver
    'name': context.properties['infra_id'] + '-api-internal-health-check',
    'type': 'compute.v1.healthCheck',
    'properties': {
      'httpsHealthCheck': {
        'port': 6443,
        'requestPath': '/readyz'
      },
      'type': "HTTPS"
    }
  }, {
    'name': context.properties['infra_id'] + '-api-internal-backend-service',
    'type': 'compute.v1.regionBackendService',
    'properties': {
      'backends': backends,
      'healthChecks': ['$$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
      'loadBalancingScheme': 'INTERNAL',
      'region': context.properties['region'],
      'protocol': 'TCP',
      'timeoutSec': 120
    }
  }, {
    'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
    'type': 'compute.v1.forwardingRule',
    'properties': {
      'backendService': '$$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
      'IPAddress': '$$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
      'loadBalancingScheme': 'INTERNAL',
      'ports': ['6443','22623'],
      'region': context.properties['region'],
      'subnetwork': context.properties['control_subnet']
    }
  }
]]

for zone in context.properties['zones']:
  resources.append({
    'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
      'namedPorts': [
        {
          'name': 'ignition',
          'port': 22623
        }, {
          'name': 'https',
          'port': 6443
        }
      ]
    }
  })

```

```

    ],
    'network': context.properties['cluster_network'],
    'zone': zone
  }
})

return {'resources': resources}

```

You will need this template in addition to the **02_lb_ext.py** template when you create an external cluster.

9.11.11. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the private DNS** section of this topic and save it as **02_dns.py** on your computer. This template describes the private DNS objects that your cluster requires.
2. Create a **02_dns.yaml** resource definition file:

```

$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' 1
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
EOF

```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **cluster_domain** is the domain for the cluster, for example **openshift.example.com**.
- 3 **cluster_network** is the **selfLink** URL to the cluster network.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

a. Add the internal DNS entries:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

b. For an external cluster, also add the external DNS entries:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

9.11.11.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

Example 9.52. 02_dns.py Deployment Manager template

```
def GenerateConfig(context):
    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.',
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    ]
```

```

    }
  }
}

return {'resources': resources}

```

9.11.12. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for firewall rules** section of this topic and save it as **03_firewall.py** on your computer. This template describes the security groups that your cluster requires.
2. Create a **03_firewall.yaml** resource definition file:

```

$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' 1
    infra_id: '${INFRA_ID}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
    network_cidr: '${NETWORK_CIDR}' 4
EOF

```

1 **allowed_external_cidr** is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to **\${NETWORK_CIDR}**.

2 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

- 3 **cluster_network** is the **selfLink** URL to the cluster network.
- 4 **network_cidr** is the CIDR of the VPC network, for example **10.0.0.0/16**.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml
```

9.11.12.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

Example 9.53. 03_firewall.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    ], {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    ]
}
```

```

}, {
  'name': context.properties['infra_id'] + '-etcd',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['2379-2380']
    }],
    'sourceTags': [context.properties['infra_id'] + '-master'],
    'targetTags': [context.properties['infra_id'] + '-master']
  }
}, {
  'name': context.properties['infra_id'] + '-control-plane',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['10257']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['10259']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['22623']
    }],
    'sourceTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ],
    'targetTags': [context.properties['infra_id'] + '-master']
  }
}, {
  'name': context.properties['infra_id'] + '-internal-network',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'icmp'
    }, {
      'IPProtocol': 'tcp',
      'ports': ['22']
    }],
    'sourceRanges': [context.properties['network_cidr']],
    'targetTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ]
  }
}, {
  'name': context.properties['infra_id'] + '-internal-cluster',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{

```



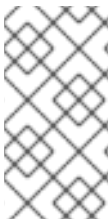
```

        'IPProtocol': 'udp',
        'ports': ['4789', '6081']
    },{
        'IPProtocol': 'udp',
        'ports': ['500', '4500']
    },{
        'IPProtocol': 'esp',
    },{
        'IPProtocol': 'tcp',
        'ports': ['9000-9999']
    },{
        'IPProtocol': 'udp',
        'ports': ['9000-9999']
    },{
        'IPProtocol': 'tcp',
        'ports': ['10250']
    },{
        'IPProtocol': 'tcp',
        'ports': ['30000-32767']
    },{
        'IPProtocol': 'udp',
        'ports': ['30000-32767']
    }
  ],
  'sourceTags': [
    context.properties['infra_id'] + '-master',
    context.properties['infra_id'] + '-worker'
  ],
  'targetTags': [
    context.properties['infra_id'] + '-master',
    context.properties['infra_id'] + '-worker'
  ]
}
}}
return {'resources': resources}

```

9.11.13. Creating IAM roles in GCP

You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for IAM roles** section of this topic and save it as **03_iam.py** on your computer. This template describes the IAM roles that your cluster requires.
2. Create a **03_iam.yaml** resource definition file:

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' 1
EOF
```

- 1** **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. Export the variable for the master service account:

```
$ export MASTER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

5. Export the variable for the worker service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

6. Export the variable for the subnet that hosts the compute machines:

```
$ export COMPUTE_SUBNET=$(gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink')
```

7. The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
```

```
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. Create a service account key and store it locally for later use:

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.11.13.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

Example 9.54. `03_iam.py` Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }
    ]

    return {'resources': resources}
```

9.11.14. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.

Procedure

1. Obtain the RHCOS image from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The file name contains the OpenShift Container Platform version number in the format **rhcos-
<version>-<arch>-gcp.<arch>.tar.gz**.

2. Create the Google storage bucket:

```
$ gsutil mb gs://<bucket_name>
```

3. Upload the RHCOS image to the Google storage bucket:

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz  
gs://<bucket_name>
```

4. Export the uploaded RHCOS image location as a variable:

```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. Create the cluster image:

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \  
--source-uri="${IMAGE_SOURCE}"
```

9.11.15. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.

- Ensure pyOpenSSL is installed.

Procedure

1. Copy the template from the **Deployment Manager template for the bootstrap machine** section of this topic and save it as **04_bootstrap.py** on your computer. This template describes the bootstrap machine that your cluster requires.
2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --
format json | jq -r .selfLink`)
```

3. Create a bucket and upload the **bootstrap.ign** file:

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-
bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. Create a **04_bootstrap.yaml** resource definition file:

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    zone: '${ZONE_0}' 3

    cluster_network: '${CLUSTER_NETWORK}' 4
    control_subnet: '${CONTROL_SUBNET}' 5
    image: '${CLUSTER_IMAGE}' 6
    machine_type: 'n1-standard-4' 7
    root_volume_size: '128' 8

    bootstrap_ign: '${BOOTSTRAP_IGN}' 9
EOF
```

1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

2 **region** is the region to deploy the cluster into, for example **us-central1**.

- 3 **zone** is the zone to deploy the bootstrap instance into, for example **us-central1-b**.
- 4 **cluster_network** is the **selfLink** URL to the cluster network.
- 5 **control_subnet** is the **selfLink** URL to the control subnet.
- 6 **image** is the **selfLink** URL to the RHCOS image.
- 7 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 8 **root_volume_size** is the boot disk size for the bootstrap machine.
- 9 **bootstrap_ign** is the URL output when creating a signed URL.

6. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the bootstrap machine manually.

a. Add the bootstrap instance to the internal load balancer instance group:

```
$ gcloud compute instance-groups unmanaged add-instances \
  ${INFRA_ID}-bootstrap-ig --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

b. Add the bootstrap instance group to the internal load balancer backend service:

```
$ gcloud compute backend-services add-backend \
  ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
group=${INFRA_ID}-bootstrap-ig --instance-group-zone=${ZONE_0}
```

9.11.15.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 9.55. 04_bootstrap.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
```

```

        'boot': True,
        'initializeParams': {
            'diskSizeGb': context.properties['root_volume_size'],
            'sourceImage': context.properties['image']
        }
    }],
    'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
    'metadata': {
        'items': [{
            'key': 'user-data',
            'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ ""},"version":"3.2.0"}}}',
        }
    ]
    },
    'networkInterfaces': [{
        'subnetwork': context.properties['control_subnet'],
        'accessConfigs': [{
            'natIP': '${ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address}'
        }
    ]
    }],
    'tags': {
        'items': [
            context.properties['infra_id'] + '-master',
            context.properties['infra_id'] + '-bootstrap'
        ]
    },
    'zone': context.properties['zone']
}
}, {
    'name': context.properties['infra_id'] + '-bootstrap-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}
]]

return {'resources': resources}

```

9.11.16. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the **Deployment Manager template for control plane machines** section of this topic and save it as **05_control_plane.py** on your computer. This template describes the control plane machines that your cluster requires.
2. Export the following variable required by the resource definition:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. Create a **05_control_plane.yaml** resource definition file:

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' 1
    zones: 2
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
    image: '${CLUSTER_IMAGE}' 4
    machine_type: 'n1-standard-4' 5
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

    ignition: '${MASTER_IGNITION}' 7
EOF
```


-
- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **zones** are the zones to deploy the control plane instances into, for example **us-central1-a**, **us-central1-b**, and **us-central1-c**.
- 3 **control_subnet** is the **selfLink** URL to the control subnet.
- 4 **image** is the **selfLink** URL to the RHCOS image.
- 5 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 6 **service_account_email** is the email address for the master service account that you created.
- 7 **ignition** is the contents of the **master.ign** file.

4. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

- Run the following commands to add the control plane machines to the appropriate instance groups:

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

- For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.11.16.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 9.56. 05_control_plane.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }
        ],
        'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }
        ]
    },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][0]
    }
    ], {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }
        ],
        'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
```

```

context.properties['machine_type'],
  'metadata': {
    'items': [{
      'key': 'user-data',
      'value': context.properties['ignition']
    }]
  },
  'networkInterfaces': [{
    'subnetwork': context.properties['control_subnet']
  }],
  'serviceAccounts': [{
    'email': context.properties['service_account_email'],
    'scopes': ['https://www.googleapis.com/auth/cloud-platform']
  }],
  'tags': {
    'items': [
      context.properties['infra_id'] + '-master',
    ]
  },
  'zone': context.properties['zones'][1]
}
}, {
  'name': context.properties['infra_id'] + '-master-2',
  'type': 'compute.v1.instance',
  'properties': {
    'disks': [{
      'autoDelete': True,
      'boot': True,
      'initializeParams': {
        'diskSizeGb': context.properties['root_volume_size'],
        'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
        'sourceImage': context.properties['image']
      }
    }],
    'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
    'metadata': {
      'items': [{
        'key': 'user-data',
        'value': context.properties['ignition']
      }]
    },
    'networkInterfaces': [{
      'subnetwork': context.properties['control_subnet']
    }],
    'serviceAccounts': [{
      'email': context.properties['service_account_email'],
      'scopes': ['https://www.googleapis.com/auth/cloud-platform']
    }],
    'tags': {
      'items': [
        context.properties['infra_id'] + '-master',
      ]
    },
    'zone': context.properties['zones'][2]
  }
}

```

```

    ]]
    return {'resources': resources}

```

9.11.17. Wait for bootstrap completion and remove bootstrap resources in GCP

After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level info 2

```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```

$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}

```

```

$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign

```

```

$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition

```

```

$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap

```

9.11.18. Creating additional worker machines in GCP

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type **06_worker.py** in the file.



NOTE

If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the **Deployment Manager template for worker machines** section of this topic and save it as **06_worker.py** on your computer. This template describes the worker machines that your cluster requires.
2. Export the variables that the resource definition uses.
 - a. Export the subnet that hosts the compute machines:

```
$ export COMPUTE_SUBNET=$(gcloud compute networks subnets describe
${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r '.selfLink')
```

- b. Export the email address for your service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '[0].email')
```

- c. Export the location of the compute machine Ignition config file:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. Create a **06_worker.yaml** resource definition file:

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' ❶
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' ❷
    zone: '${ZONE_0}' ❸
    compute_subnet: '${COMPUTE_SUBNET}' ❹
    image: '${CLUSTER_IMAGE}' ❺
    machine_type: 'n1-standard-4' ❻
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' ❼
    ignition: '${WORKER_IGNITION}' ❽
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' ❾
    zone: '${ZONE_1}' ❿
    compute_subnet: '${COMPUTE_SUBNET}' ⓫
    image: '${CLUSTER_IMAGE}' ⓬
    machine_type: 'n1-standard-4' ⓭
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' ⓮
    ignition: '${WORKER_IGNITION}' ⓯
EOF
```

- ❶ **name** is the name of the worker machine, for example **worker-0**.
- ❷ ❾ **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- ❸ ❿ **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.
- ❹ ⓫ **compute_subnet** is the **selfLink** URL to the compute subnet.
- ❺ ⓬ **image** is the **selfLink** URL to the RHCOS image. ¹
- ❻ ⓭ **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- ❼ ⓮ **service_account_email** is the email address for the worker service account that you created.
- ❽ ⓯ **ignition** is the contents of the **worker.ign** file.

4. Optional: If you want to launch additional instances, include additional resources of type **06_worker.py** in your **06_worker.yaml** resource definition file.
5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

- To use a GCP Marketplace image, specify the offer to use:
 - OpenShift Container Platform: <https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-48-x86-64-202210040145>
 - OpenShift Platform Plus: <https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-48-x86-64-202206140145>
 - OpenShift Kubernetes Engine: <https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-48-x86-64-202206140145>

9.11.18.1. Deployment Manager template for worker machines

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 9.57. 06_worker.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
```

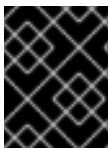
```

    ]
  },
  'zone': context.properties['zone']
}
}}
return {'resources': resources}

```

9.11.19. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

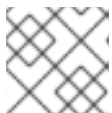
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.11.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The

kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

9.11.21. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE  VERSION
master-0  Ready    master   63m  v1.25.0
master-1  Ready    master   63m  v1.25.0
master-2  Ready    master   64m  v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

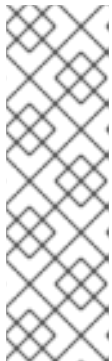
```
$ oc get csr
```

Example output

```
NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n}} | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n}} | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   73m   v1.25.0
master-1  Ready   master   73m   v1.25.0
```

```

master-2 Ready   master 74m v1.25.0
worker-0 Ready   worker 11m v1.25.0
worker-1 Ready   worker 11m v1.25.0

```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

9.11.22. Optional: Adding the ingress DNS records

If you removed the DNS zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- Configure a GCP account.
- Remove the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

Procedure

1. Wait for the Ingress router to create a load balancer and populate the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```

NAME           TYPE           CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
router-default LoadBalancer   172.30.18.154 35.233.157.184 80:32288/TCP,443:31215/TCP 98

```

2. Add the A record to your zones:

- To use A records:

- i. Export the variable for the router IP address:

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

- ii. Add the A record to the private zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

- iii. For an external cluster, also add the A record to the public zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

- To add explicit domains instead of using a wildcard, create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host} {"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

9.11.23. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.
- Install the **oc** CLI and log in.

Procedure

1. Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Observe the running state of your cluster.

- a. Run the following command to view the current cluster version and status:

```
$ oc get clusterversion
```

Example output

```
NAME      VERSION  AVAILABLE  PROGRESSING  SINCE  STATUS
version   False    True        24m          Working towards 4.5.4: 99% complete
```

- b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

```
$ oc get clusteroperators
```

Example output

```
NAME              VERSION  AVAILABLE  PROGRESSING  DEGRADED  SINCE
authentication    4.5.4   True       False        False     7m56s
cloud-credential   4.5.4   True       False        False     31m
cluster-autoscaler 4.5.4   True       False        False     16m
console            4.5.4   True       False        False     10m
csi-snapshot-controller 4.5.4   True       False        False     16m
```

dns	4.5.4	True	False	False	22m
etcd	4.5.4	False	False	False	25s
image-registry	4.5.4	True	False	False	16m
ingress	4.5.4	True	False	False	16m
insights	4.5.4	True	False	False	17m
kube-apiserver	4.5.4	True	False	False	19m
kube-controller-manager	4.5.4	True	False	False	20m
kube-scheduler	4.5.4	True	False	False	20m
kube-storage-version-migrator	4.5.4	True	False	False	16m
machine-api	4.5.4	True	False	False	22m
machine-config	4.5.4	True	False	False	22m
marketplace	4.5.4	True	False	False	16m
monitoring	4.5.4	True	False	False	10m
network	4.5.4	True	False	False	23m
node-tuning	4.5.4	True	False	False	23m
openshift-apiserver	4.5.4	True	False	False	17m
openshift-controller-manager	4.5.4	True	False	False	15m
openshift-samples	4.5.4	True	False	False	16m
operator-lifecycle-manager	4.5.4	True	False	False	22m
operator-lifecycle-manager-catalog	4.5.4	True	False	False	22m
operator-lifecycle-manager-packageserver	4.5.4	True	False	False	18m
service-ca	4.5.4	True	False	False	23m
service-catalog-apiserver	4.5.4	True	False	False	23m
service-catalog-controller-manager	4.5.4	True	False	False	23m
storage	4.5.4	True	False	False	17m

- c. Run the following command to view your cluster pods:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE                               NAME
READY  STATUS  RESTARTS  AGE
kube-system                               etcd-member-ip-10-0-3-111.us-east-
2.compute.internal                       1/1    Running  0    35m
kube-system                               etcd-member-ip-10-0-3-239.us-east-
2.compute.internal                       1/1    Running  0    37m
kube-system                               etcd-member-ip-10-0-3-24.us-east-
2.compute.internal                       1/1    Running  0    35m
openshift-apiserver-operator              openshift-apiserver-operator-6d6674f4f4-
h7t2t                                     1/1    Running  1    37m
openshift-apiserver                       apiserver-fm48r
1/1    Running  0    30m
openshift-apiserver                       apiserver-fxkvv
1/1    Running  0    29m
openshift-apiserver                       apiserver-q85nm
1/1    Running  0    29m
...
openshift-service-ca-operator             openshift-service-ca-operator-66ff6dc6cd-
9r257                                     1/1    Running  0    37m
openshift-service-ca                       apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1    Running  0    35m
openshift-service-ca                       configmap-cabundle-injector-8498544d7-
25qn6                                     1/1    Running  0    35m

```



```

openshift-service-ca                                service-serving-cert-signer-6445fc9c6-wqdqn
1/1    Running    0        35m
openshift-service-catalog-apiserver-operator        openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w    1/1    Running    0        32m
openshift-service-catalog-controller-manager-operator openshift-service-catalog-
controller-manager-operator-b78cr2lnm    1/1    Running    0        31m

```

When the current cluster version is **AVAILABLE**, the installation is complete.

9.11.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.11.25. Next steps

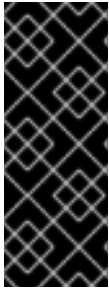
- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- [Configure Global Access for an Ingress Controller on GCP](#).

9.12. INSTALLING A CLUSTER INTO A SHARED VPC ON GCP USING DEPLOYMENT MANAGER TEMPLATES

In OpenShift Container Platform version 4.12, you can install a cluster into a shared Virtual Private Cloud (VPC) on Google Cloud Platform (GCP) that uses infrastructure that you provide. In this context, a cluster installed into a shared VPC is a cluster that is configured to use a VPC from a project different from where the cluster is being deployed.

A shared VPC enables an organization to connect resources from multiple projects to a common VPC network. You can communicate within the organization securely and efficiently by using internal IPs from that network. For more information about shared VPC, see [Shared VPC overview](#) in the GCP documentation.

The steps for performing a user-provided infrastructure installation into a shared VPC are outlined here. Several [Deployment Manager](#) templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.

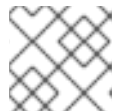


IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

9.12.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).



NOTE

Be sure to also review this site list if you are configuring a proxy.

9.12.2. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

9.12.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.12.4. Configuring the GCP project that hosts your cluster

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

9.12.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See [Creating and Managing Projects](#) in the GCP documentation.



IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the **api-int.<cluster_name>.<base_domain>** URL; the Premium Tier is required for internal load balancing.

9.12.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. You may also enable optional API services which are not required for installation. See [Enabling services](#) in the GCP documentation.

Table 9.50. Required API services

API service	Console service name
Compute Engine API	compute.googleapis.com

API service	Console service name
Cloud Resource Manager API	cloudfirewall.googleapis.com
Google DNS API	dns.googleapis.com
IAM Service Account Credentials API	iamcredentials.googleapis.com
Identity and Access Management (IAM) API	iam.googleapis.com
Service Usage API	serviceusage.googleapis.com

Table 9.51. Optional API services

API service	Console service name
Cloud Deployment Manager V2 API	deploymentmanager.googleapis.com
Google Cloud APIs	cloudapis.googleapis.com
Service Management API	servicemanagement.googleapis.com
Google Cloud Storage JSON API	storage-api.googleapis.com
Cloud Storage	storage-component.googleapis.com

9.12.4.3. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default [Quotas](#) do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

Table 9.52. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	6	1
Firewall rules	Networking	Global	11	1
Forwarding rules	Compute	Global	2	0

Service	Component	Location	Total resources required	Resources removed after bootstrap
Health checks	Compute	Global	2	0
Images	Compute	Global	1	0
Networks	Networking	Global	1	0
Routers	Networking	Global	1	0
Routes	Networking	Global	2	0
Subnetworks	Compute	Global	2	0
Target pools	Networking	Global	2	0



NOTE

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europa-north1**
- **europa-west2**
- **europa-west3**
- **europa-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

You can increase resource quotas from the [GCP console](#), but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

9.12.4.4. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See [Creating a service account](#) in the GCP documentation.
2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See [Granting roles to a service account for specific resources](#).



NOTE

While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. You can create the service account key in JSON format, or attach the service account to a GCP virtual machine. See [Creating service account keys](#) and [Creating and enabling service accounts for instances](#) in the GCP documentation.
You must have a service account key or a virtual machine with an attached service account to create the cluster.



NOTE

If you use a virtual machine with an attached service account to create your cluster, you must set **credentialsMode: Manual** in the **install-config.yaml** file before installation.

9.12.4.4.1. Required GCP roles

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. If the security policies for your organization require a more restrictive set of permissions, you can create a service account with the following permissions.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions.

If you deploy your cluster into an existing virtual private cloud (VPC), the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- IAM Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Required roles for using passthrough credentials mode

- Compute Load Balancer Admin
- IAM Role Viewer

Required roles for user-provisioned GCP infrastructure

- Deployment Manager Editor

The roles are applied to the service accounts that the control plane and compute machines use:

Table 9.53. GCP service account permissions

Account	Roles
Control Plane	roles/compute.instanceAdmin
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	roles/compute.viewer
	roles/storage.admin

9.12.4.5. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **australia-southeast1** (Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europa-central2** (Warsaw, Poland)
- **europa-north1** (Hamina, Finland)
- **europa-southwest1** (Madrid, Spain)
- **europa-west1** (St. Ghislain, Belgium)
- **europa-west2** (London, England, UK)
- **europa-west3** (Frankfurt, Germany)
- **europa-west4** (Eemshaven, Netherlands)
- **europa-west6** (Zürich, Switzerland)
- **europa-west8** (Milan, Italy)
- **europa-west9** (Paris, France)
- **europa-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)

- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)



NOTE

To determine which machine type instances are available by region and zone, see the Google [documentation](#).

9.12.4.6. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

Prerequisites

- You created a project to host your cluster.
- You created a service account and granted it the required permissions.

Procedure

1. Install the following binaries in **\$PATH**:

- **gcloud**
- **gsutil**

See [Install the latest Cloud SDK version](#) in the GCP documentation.

2. Authenticate using the **gcloud** tool with your configured service account.
See [Authorizing with a service account](#) in the GCP documentation.

9.12.5. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

9.12.5.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 9.54. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

9.12.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.55. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

9.12.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.58. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.12.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

9.12.6. Configuring the GCP project that hosts your shared VPC network

If you use a shared Virtual Private Cloud (VPC) to host your OpenShift Container Platform cluster in Google Cloud Platform (GCP), you must configure the project that hosts it.



NOTE

If you already have a project that hosts the shared VPC network, review this section to ensure that the project meets all of the requirements to install an OpenShift Container Platform cluster.

Procedure

1. Create a project to host the shared VPC for your OpenShift Container Platform cluster. See [Creating and Managing Projects](#) in the GCP documentation.
2. Create a service account in the project that hosts your shared VPC. See [Creating a service account](#) in the GCP documentation.
3. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See [Granting roles to a service account for specific resources](#).



NOTE

While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

The service account for the project that hosts the shared VPC network requires the following roles:

- Compute Network User
- Compute Security Admin
- Deployment Manager Editor
- DNS Administrator
- Security Admin
- Network Management Admin

9.12.6.1. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the project that hosts the shared VPC that you install the cluster into. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.



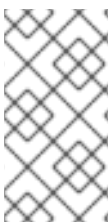
NOTE

If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](#).

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](#) in the GCP documentation. Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](#) in the GCP documentation. You typically have four name servers.
4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](#).
5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](#) in the GCP documentation.
6. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain. This process might include a request to your company's IT department or the division that controls the root domain and DNS services for your company.

9.12.6.2. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.

Procedure

1. Copy the template from the **Deployment Manager template for the VPC** section of this topic and save it as **01_vpc.py** on your computer. This template describes the VPC that your cluster requires.
2. Export the following variables required by the resource definition:

- a. Export the control plane CIDR:

```
$ export MASTER_SUBNET_CIDR='10.0.0.0/17'
```

- b. Export the compute CIDR:

```
$ export WORKER_SUBNET_CIDR='10.0.128.0/17'
```

- c. Export the region to deploy the VPC network and cluster to:

```
$ export REGION='<region>'
```

3. Export the variable for the ID of the project that hosts the shared VPC:

```
$ export HOST_PROJECT=<host_project>
```

4. Export the variable for the email of the service account that belongs to host project:

```
$ export HOST_PROJECT_ACCOUNT=<host_service_account_email>
```

5. Create a **01_vpc.yaml** resource definition file:

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '<prefix>' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

- 1 **infra_id** is the prefix of the network name.
- 2 **region** is the region to deploy the cluster into, for example **us-central1**.
- 3 **master_subnet_cidr** is the CIDR for the master subnet, for example **10.0.0.0/17**.
- 4 **worker_subnet_cidr** is the CIDR for the worker subnet, for example **10.0.128.0/17**.

6. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create <vpc_deployment_name> --config
01_vpc.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} 1
```

1 For **<vpc_deployment_name>**, specify the name of the VPC to deploy.

7. Export the VPC variable that other components require:

a. Export the name of the host project network:

```
$ export HOST_PROJECT_NETWORK=<vpc_network>
```

b. Export the name of the host project control plane subnet:

```
$ export HOST_PROJECT_CONTROL_SUBNET=<control_plane_subnet>
```

c. Export the name of the host project compute subnet:

```
$ export HOST_PROJECT_COMPUTE_SUBNET=<compute_subnet>
```

8. Set up the shared VPC. See [Setting up Shared VPC](#) in the GCP documentation.

9.12.6.2.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

Example 9.59. 01_vpc.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }
]
```

```

    }, {
      'name': context.properties['infra_id'] + '-router',
      'type': 'compute.v1.router',
      'properties': {
        'region': context.properties['region'],
        'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
        'nats': [{
          'name': context.properties['infra_id'] + '-nat-master',
          'natIpAllocateOption': 'AUTO_ONLY',
          'minPortsPerVm': 7168,
          'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
          'subnetworks': [{
            'name': '${ref.' + context.properties['infra_id'] + '-master-subnet.selfLink}',
            'sourceIpRangesToNat': ['ALL_IP_RANGES']
          }]
        }]
      }, {
        'name': context.properties['infra_id'] + '-nat-worker',
        'natIpAllocateOption': 'AUTO_ONLY',
        'minPortsPerVm': 512,
        'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
        'subnetworks': [{
          'name': '${ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink}',
          'sourceIpRangesToNat': ['ALL_IP_RANGES']
        }]
      }
    ]
  }
}

return {'resources': resources}

```

9.12.7. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

9.12.7.1. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:


```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

9.12.7.2. Sample customized **install-config.yaml** file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      tags: 5
      - control-plane-tag1
      - control-plane-tag2
  replicas: 3
compute: 6
```

```

- hyperthreading: Enabled 7
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
      tags: 8
        - compute-tag1
        - compute-tag2
    replicas: 0
  metadata:
    name: test-cluster
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
      - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 9
    serviceNetwork:
      - 172.30.0.0/16
  platform:
    gcp:
      defaultMachinePlatform:
        tags: 10
          - global-tag1
          - global-tag2
      projectID: openshift-production 11
      region: us-central1 12
    pullSecret: '{"auths": ...}'
    fips: false 13
    sshKey: ssh-ed25519 AAAA... 14
    publish: Internal 15

```

- 1 Specify the public DNS on the host project.
- 2 6 If you do not provide these parameters and values, the installation program provides the default value.
- 3 7 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 4 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

- 5 8 10 Optional: A set of network tags to apply to the control plane or compute machine sets. The **platform.gcp.defaultMachinePlatform.tags** parameter applies to both control plane and compute machines. If the **compute.platform.gcp.tags** or **controlPlane.platform.gcp.tags** parameters are set, they override the **platform.gcp.defaultMachinePlatform.tags** parameter.
- 9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 11 Specify the main project where the VM instances reside.
- 12 Specify the region that your VPC network is in.
- 13 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 14 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 15 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**. To use a shared VPC in a cluster that uses infrastructure that you provision, you must set **publish** to **Internal**. The installation program will no longer be able to access the public DNS zone for the base domain in the host project.

9.12.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

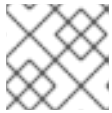
Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5

Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.12.7.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

- a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
- b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
- c. Save and exit the file.

5. Remove the **privateZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
status: {}
```

- 1** Remove this section completely.

6. Configure the cloud provider for your VPC.
 - a. Open the `<installation_directory>/manifests/cloud-provider-config.yaml` file.
 - b. Add the **network-project-id** parameter and set its value to the ID of project that hosts the shared VPC network.
 - c. Add the **network-name** parameter and set its value to the name of the shared VPC network that hosts the OpenShift Container Platform cluster.
 - d. Replace the value of the **subnetwork-name** parameter with the value of the shared VPC subnet that hosts your compute machines.

The contents of the `<installation_directory>/manifests/cloud-provider-config.yaml` resemble the following example:

```
config: |+
  [global]
  project-id    = example-project
  regional     = true
  multizone    = true
  node-tags    = opensh-ptzzx-master
  node-tags    = opensh-ptzzx-worker
  node-instance-prefix = opensh-ptzzx
  external-instance-groups-prefix = opensh-ptzzx
  network-project-id = example-shared-vpc
  network-name  = example-network
  subnetwork-name = example-worker-subnet
```

7. If you deploy a cluster that is not on a private network, open the `<installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml` file and replace the value of the **scope** parameter with **External**. The contents of the file resemble the following example:

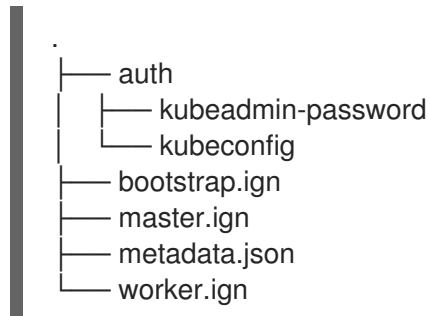
```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
      type: LoadBalancerService
status:
  availableReplicas: 0
  domain: ""
  selector: ""
```

8. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The `kubeadmin-password` and `kubeconfig` files are created in the `./<installation_directory>/auth` directory:



9.12.8. Exporting common variables

9.12.8.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

9.12.8.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).



NOTE

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Generate the Ignition config files for your cluster.
- Install the **jq** package.

Procedure

1. Export the following common variables to be used by the provided Deployment Manager templates:

```
$ export BASE_DOMAIN='<base_domain>' 1
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>' 2
$ export NETWORK_CIDR='10.0.0.0/16'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 3
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
```

- 1 2 Supply the values for the host project.

- 3 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

9.12.9. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

9.12.9.1. Setting the cluster node hostnames through DHCP

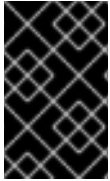
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

9.12.9.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 9.56. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 9.57. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 9.58. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

9.12.10. Creating load balancers in GCP

You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the internal load balancer** section of this topic and save it as **02_lb_int.py** on your computer. This template describes the internal load balancing objects that your cluster requires.
2. For an external cluster, also copy the template from the **Deployment Manager template for the external load balancer** section of this topic and save it as **02_lb_ext.py** on your computer. This template describes the external load balancing objects that your cluster requires.
3. Export the variables that the deployment template uses:
 - a. Export the cluster network location:

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe
${HOST_PROJECT_NETWORK} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT} --format json | jq -r .selfLink`)
```

- b. Export the control plane subnet location:

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe
${HOST_PROJECT_CONTROL_SUBNET} --region=${REGION} --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
.selfLink`)
```

- c. Export the three zones that the cluster uses:

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[2] | cut -d "/" -f9`)
```

4. Create a **02_infra.yaml** resource definition file:

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py 1
resources:
- name: cluster-lb-ext 2
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' 3
    region: '${REGION}' 4
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' 5
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: 6
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

1 **2** Required only when deploying an external cluster.

3 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

4 **region** is the region to deploy the cluster into, for example **us-central1**.

5 **control_subnet** is the URI to the control subnet.

6 **zones** are the zones to deploy the control plane instances into, like **us-east1-b**, **us-east1-c**, and **us-east1-d**.

5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:

```
$ export CLUSTER_IP=`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --region=${REGION} --format json | jq -r .address`
```

7. For an external cluster, also export the cluster public IP address:

```
$ export CLUSTER_PUBLIC_IP=`gcloud compute addresses describe ${INFRA_ID}-cluster-public-ip --region=${REGION} --format json | jq -r .address`
```

9.12.10.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

Example 9.60. `02_lb_ext.py` Deployment Manager template

```
def GenerateConfig(context):
    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
        # probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$${ref.} + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '${ref.} + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '${ref.} + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
```

```

    }
  }}
  return {'resources': resources}

```

9.12.10.2. Deployment Manager template for the internal load balancer

You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

Example 9.61. 02_ib_int.py Deployment Manager template

```

def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
        # probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',

```

```

        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '${ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink}',
            'IPAddress': '${ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink}',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    ]
}

for zone in context.properties['zones']:
    resources.append({
        'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namedPorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                }, {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': zone
        }
    })

return {'resources': resources}

```

You will need this template in addition to the **02_lb_ext.py** template when you create an external cluster.

9.12.11. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the private DNS** section of this topic and save it as **02_dns.py** on your computer. This template describes the private DNS objects that your cluster requires.
2. Create a **02_dns.yaml** resource definition file:

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' 1
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
EOF
```

- 1** **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2** **cluster_domain** is the domain for the cluster, for example **openshift.example.com**.
- 3** **cluster_network** is the **selfLink** URL to the cluster network.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:
 - a. Add the internal DNS entries:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

- b. For an external cluster, also add the external DNS entries:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
```



```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

9.12.11.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

Example 9.62. `02_dns.py` Deployment Manager template

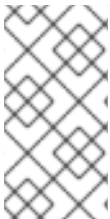
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.',
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    }]

    return {'resources': resources}
```

9.12.12. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for firewall rules** section of this topic and save it as **03_firewall.py** on your computer. This template describes the security groups that your cluster requires.
2. Create a **03_firewall.yaml** resource definition file:

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' 1
    infra_id: '${INFRA_ID}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
    network_cidr: '${NETWORK_CIDR}' 4
EOF
```

- 1 **allowed_external_cidr** is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to **\${NETWORK_CIDR}**.
- 2 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 3 **cluster_network** is the **selfLink** URL to the cluster network.
- 4 **network_cidr** is the CIDR of the VPC network, for example **10.0.0.0/16**.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

9.12.12.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

Example 9.63. 03_firewall.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
```

```

        'IPProtocol': 'tcp',
        'ports': ['22']
    }],
    'sourceRanges': [context.properties['allowed_external_cidr']],
    'targetTags': [context.properties['infra_id'] + '-bootstrap']
}
}, {
    'name': context.properties['infra_id'] + '-api',
    'type': 'compute.v1.firewall',
    'properties': {
        'network': context.properties['cluster_network'],
        'allowed': [{
            'IPProtocol': 'tcp',
            'ports': ['6443']
        }],
        'sourceRanges': [context.properties['allowed_external_cidr']],
        'targetTags': [context.properties['infra_id'] + '-master']
    }
}, {
    'name': context.properties['infra_id'] + '-health-checks',
    'type': 'compute.v1.firewall',
    'properties': {
        'network': context.properties['cluster_network'],
        'allowed': [{
            'IPProtocol': 'tcp',
            'ports': ['6080', '6443', '22624']
        }],
        'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
        'targetTags': [context.properties['infra_id'] + '-master']
    }
}, {
    'name': context.properties['infra_id'] + '-etcd',
    'type': 'compute.v1.firewall',
    'properties': {
        'network': context.properties['cluster_network'],
        'allowed': [{
            'IPProtocol': 'tcp',
            'ports': ['2379-2380']
        }],
        'sourceTags': [context.properties['infra_id'] + '-master'],
        'targetTags': [context.properties['infra_id'] + '-master']
    }
}, {
    'name': context.properties['infra_id'] + '-control-plane',
    'type': 'compute.v1.firewall',
    'properties': {
        'network': context.properties['cluster_network'],
        'allowed': [{
            'IPProtocol': 'tcp',
            'ports': ['10257']
        }],
        'IPProtocol': 'tcp',
        'ports': ['10259']
    }, {
        'IPProtocol': 'tcp',
        'ports': ['22623']
    }
}
}

```

```

    }],
    'sourceTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ],
    'targetTags': [context.properties['infra_id'] + '-master']
  }
}, {
  'name': context.properties['infra_id'] + '-internal-network',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'icmp'
    }, {
      'IPProtocol': 'tcp',
      'ports': ['22']
    }],
    'sourceRanges': [context.properties['network_cidr']],
    'targetTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ]
  }
}, {
  'name': context.properties['infra_id'] + '-internal-cluster',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'udp',
      'ports': ['4789', '6081']
    }, {
      'IPProtocol': 'udp',
      'ports': ['500', '4500']
    }, {
      'IPProtocol': 'esp',
    }, {
      'IPProtocol': 'tcp',
      'ports': ['9000-9999']
    }, {
      'IPProtocol': 'udp',
      'ports': ['9000-9999']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['10250']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['30000-32767']
    }, {
      'IPProtocol': 'udp',
      'ports': ['30000-32767']
    }],
    'sourceTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ]
  }
}

```

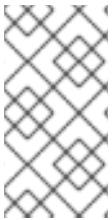
```

    ],
    'targetTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ]
  }
}
}
return {'resources': resources}

```

9.12.13. Creating IAM roles in GCP

You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for IAM roles** section of this topic and save it as **03_iam.py** on your computer. This template describes the IAM roles that your cluster requires.
2. Create a **03_iam.yaml** resource definition file:

```

$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' 1
EOF

```

- 1** **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

3. Create the deployment by using the **gcloud** CLI:

-

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. Export the variable for the master service account:

```
$ export MASTER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '[0].email')
```

5. Export the variable for the worker service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '[0].email')
```

6. Assign the permissions that the installation program requires to the service accounts for the subnets that host the control plane and compute subnets:

- a. Grant the **networkViewer** role of the project that hosts your shared VPC to the master service account:

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
projects add-iam-policy-binding ${HOST_PROJECT} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role
"roles/compute.networkViewer"
```

- b. Grant the **networkUser** role to the master service account for the control plane subnet:

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

- c. Grant the **networkUser** role to the worker service account for the control plane subnet:

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

- d. Grant the **networkUser** role to the master service account for the compute subnet:

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

- e. Grant the **networkUser** role to the worker service account for the compute subnet:

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
```

```
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

- The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

- Create a service account key and store it locally for later use:

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.12.13.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

Example 9.64. 03_iam.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }
    ]

    return {'resources': resources}
```

9.12.14. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.

Procedure

1. Obtain the RHCOS image from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The file name contains the OpenShift Container Platform version number in the format **rhcos-
<version>-<arch>-gcp.<arch>.tar.gz**.

2. Create the Google storage bucket:

```
$ gsutil mb gs://<bucket_name>
```

3. Upload the RHCOS image to the Google storage bucket:

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz  
gs://<bucket_name>
```

4. Export the uploaded RHCOS image location as a variable:

```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. Create the cluster image:

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \  
--source-uri="${IMAGE_SOURCE}"
```

9.12.15. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Ensure pyOpenSSL is installed.

Procedure

1. Copy the template from the **Deployment Manager template for the bootstrap machine** section of this topic and save it as **04_bootstrap.py** on your computer. This template describes the bootstrap machine that your cluster requires.
2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:

```
$ export CLUSTER_IMAGE=$(gcloud compute images describe ${INFRA_ID}-rhcos-image --
format json | jq -r .selfLink`)
```

3. Create a bucket and upload the **bootstrap.ign** file:

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-
bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}`
```

5. Create a **04_bootstrap.yaml** resource definition file:

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    zone: '${ZONE_0}' 3

    cluster_network: '${CLUSTER_NETWORK}' 4
    control_subnet: '${CONTROL_SUBNET}' 5
    image: '${CLUSTER_IMAGE}' 6
```

```

machine_type: 'n1-standard-4' 7
root_volume_size: '128' 8

bootstrap_ign: '${BOOTSTRAP_IGN}' 9
EOF

```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **region** is the region to deploy the cluster into, for example **us-central1**.
- 3 **zone** is the zone to deploy the bootstrap instance into, for example **us-central1-b**.
- 4 **cluster_network** is the **selfLink** URL to the cluster network.
- 5 **control_subnet** is the **selfLink** URL to the control subnet.
- 6 **image** is the **selfLink** URL to the RHCOS image.
- 7 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 8 **root_volume_size** is the boot disk size for the bootstrap machine.
- 9 **bootstrap_ign** is the URL output when creating a signed URL.

6. Create the deployment by using the **gcloud** CLI:

```

$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml

```

7. Add the bootstrap instance to the internal load balancer instance group:

```

$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-bootstrap-ig --
zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap

```

8. Add the bootstrap instance group to the internal load balancer backend service:

```

$ gcloud compute backend-services add-backend ${INFRA_ID}-api-internal-backend-service
--region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}

```

9.12.15.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 9.65. 04_bootstrap.py Deployment Manager template

```

def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {

```

```

        'region': context.properties['region']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"},"version":"3.2.0"}}}',
            }],
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet'],
            'accessConfigs': [{
                'natIP': '${ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address}'
            }],
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-bootstrap'
            ]
        },
        'zone': context.properties['zone']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}
]]

return {'resources': resources}

```

9.12.16. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the **Deployment Manager template for control plane machines** section of this topic and save it as **05_control_plane.py** on your computer. This template describes the control plane machines that your cluster requires.
2. Export the following variable required by the resource definition:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. Create a **05_control_plane.yaml** resource definition file:

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' 1
    zones: 2
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
```

```

image: '${CLUSTER_IMAGE}' 4
machine_type: 'n1-standard-4' 5
root_volume_size: '128'
service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

ignition: '${MASTER_IGNITION}' 7
EOF

```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **zones** are the zones to deploy the control plane instances into, for example **us-central1-a**, **us-central1-b**, and **us-central1-c**.
- 3 **control_subnet** is the **selfLink** URL to the control subnet.
- 4 **image** is the **selfLink** URL to the RHCOS image.
- 5 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 6 **service_account_email** is the email address for the master service account that you created.
- 7 **ignition** is the contents of the **master.ign** file.

4. Create the deployment by using the **gcloud** CLI:

```

$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml

```

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

- Run the following commands to add the control plane machines to the appropriate instance groups:

```

$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0

```

```

$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1

```

```

$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2

```

- For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

```

$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0

```

```

$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1

```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.12.16.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 9.66. `05_control_plane.py` Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    ], {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
```

```

        'diskSizeGb': context.properties['root_volume_size'],
        'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
        'sourceImage': context.properties['image']
    }
  }],
  'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
  'metadata': {
    'items': [{
      'key': 'user-data',
      'value': context.properties['ignition']
    }]
  },
  'networkInterfaces': [{
    'subnetwork': context.properties['control_subnet']
  }],
  'serviceAccounts': [{
    'email': context.properties['service_account_email'],
    'scopes': ['https://www.googleapis.com/auth/cloud-platform']
  }],
  'tags': {
    'items': [
      context.properties['infra_id'] + '-master',
    ]
  },
  'zone': context.properties['zones'][1]
}
}, {
  'name': context.properties['infra_id'] + '-master-2',
  'type': 'compute.v1.instance',
  'properties': {
    'disks': [{
      'autoDelete': True,
      'boot': True,
      'initializeParams': {
        'diskSizeGb': context.properties['root_volume_size'],
        'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
        'sourceImage': context.properties['image']
      }
    }]
  },
  'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
  'metadata': {
    'items': [{
      'key': 'user-data',
      'value': context.properties['ignition']
    }]
  },
  'networkInterfaces': [{
    'subnetwork': context.properties['control_subnet']
  }],
  'serviceAccounts': [{
    'email': context.properties['service_account_email'],
    'scopes': ['https://www.googleapis.com/auth/cloud-platform']
  }],
  'tags': {

```

```

        'items': [
            context.properties['infra_id'] + '-master',
        ]
    },
    'zone': context.properties['zones'][2]
}
}}

return {'resources': resources}

```

9.12.17. Wait for bootstrap completion and remove bootstrap resources in GCP

After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```



```
$ gsutil rm gs://{INFRA_ID}-bootstrap-ignition/bootstrap.ign
```

```
$ gsutil rb gs://{INFRA_ID}-bootstrap-ignition
```

```
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

9.12.18. Creating additional worker machines in GCP

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type **06_worker.py** in the file.



NOTE

If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the **Deployment Manager template for worker machines** section of this topic and save it as **06_worker.py** on your computer. This template describes the worker machines that your cluster requires.
2. Export the variables that the resource definition uses.
 - a. Export the subnet that hosts the compute machines:

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
${HOST_PROJECT_COMPUTE_SUBNET} --region=${REGION} --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
.selfLink`)
```

- b. Export the email address for your service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

- c. Export the location of the compute machine Ignition config file:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. Create a **06_worker.yaml** resource definition file:

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' 1
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' 2
    zone: '${ZONE_0}' 3
    compute_subnet: '${COMPUTE_SUBNET}' 4
    image: '${CLUSTER_IMAGE}' 5
    machine_type: 'n1-standard-4' 6
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 7
    ignition: '${WORKER_IGNITION}' 8
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' 9
    zone: '${ZONE_1}' 10
    compute_subnet: '${COMPUTE_SUBNET}' 11
    image: '${CLUSTER_IMAGE}' 12
    machine_type: 'n1-standard-4' 13
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 14
    ignition: '${WORKER_IGNITION}' 15
EOF
```

- 1 **name** is the name of the worker machine, for example **worker-0**.
- 2 9 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 3 10 **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.
- 4 11 **compute_subnet** is the **selfLink** URL to the compute subnet.
- 5 12 **image** is the **selfLink** URL to the RHCOS image. ¹
- 6 13 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 7 14

service_account_email is the email address for the worker service account that you created.

8 15 **ignition** is the contents of the **worker.ign** file.

4. Optional: If you want to launch additional instances, include additional resources of type **06_worker.py** in your **06_worker.yaml** resource definition file.
5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

1. To use a GCP Marketplace image, specify the offer to use:
 - OpenShift Container Platform: **<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-48-x86-64-202210040145>**
 - OpenShift Platform Plus: **<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-48-x86-64-202206140145>**
 - OpenShift Kubernetes Engine: **<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-48-x86-64-202206140145>**

9.12.18.1. Deployment Manager template for worker machines

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 9.67. **06_worker.py** Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            }
        },
    ]
```

```

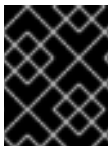
'networkInterfaces': [{
  'subnetwork': context.properties['compute_subnet']
}],
'serviceAccounts': [{
  'email': context.properties['service_account_email'],
  'scopes': ['https://www.googleapis.com/auth/cloud-platform']
}],
'tags': {
  'items': [
    context.properties['infra_id'] + '-worker',
  ]
},
'zone': context.properties['zone']
}
}]

return {'resources': resources}

```

9.12.19. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

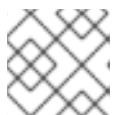
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

9.12.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

9.12.21. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

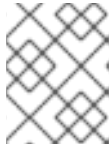
Example output

```

NAME     STATUS  ROLES  AGE  VERSION
master-0 Ready   master 63m  v1.25.0
master-1 Ready   master 63m  v1.25.0
master-2 Ready   master 64m  v1.25.0

```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```

NAME     AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...

```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

9.12.22. Adding the ingress DNS records

DNS zone configuration is removed when creating Kubernetes manifests and generating Ignition configs. You must manually create DNS records that point at the ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- Configure a GCP account.
- Remove the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

Procedure

1. Wait for the Ingress router to create a load balancer and populate the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```
NAME           TYPE           CLUSTER-IP   EXTERNAL-IP   PORT(S)           AGE
router-default LoadBalancer   172.30.18.154 35.233.157.184 80:32288/TCP,443:31215/TCP 98
```

2. Add the A record to your zones:

- To use A records:

- i. Export the variable for the router IP address:

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

- ii. Add the A record to the private zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --
project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

- iii. For an external cluster, also add the A record to the public zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME} -
-project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
```

- To add explicit domains instead of using a wildcard, create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

Example output

```

oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com

```

9.12.23. Adding ingress firewall rules

The cluster requires several firewall rules. If you do not use a shared VPC, these rules are created by the Ingress Controller via the GCP cloud provider. When you use a shared VPC, you can either create cluster-wide firewall rules for all services now or create each rule based on events, when the cluster requests access. By creating each rule when the cluster requests access, you know exactly which firewall rules are required. By creating cluster-wide firewall rules, you can apply the same rule set across multiple clusters.

If you choose to create each rule based on events, you must create firewall rules after you provision the cluster and during the life of the cluster when the console notifies you that rules are missing. Events that are similar to the following event are displayed, and you must add the firewall rules that are required:

```
$ oc get events -n openshift-ingress --field-selector="reason=LoadBalancerManualChange"
```

Example output

```

Firewall change required by security admin: `gcloud compute firewall-rules create k8s-fw-
a26e631036a3f46cba28f8df67266d55 --network example-network --description "
{"kubernetes.io/service-name":"openshift-ingress/router-default", "kubernetes.io/service-
ip":"35.237.236.234"}" --allow tcp:443,tcp:80 --source-ranges 0.0.0.0/0 --target-tags exampl-fqzq7-
master,exampl-fqzq7-worker --project example-project`

```

If you encounter issues when creating these rule-based events, you can configure the cluster-wide firewall rules while your cluster is running.

9.12.23.1. Creating cluster-wide firewall rules for a shared VPC in GCP

You can create cluster-wide firewall rules to allow the access that the OpenShift Container Platform cluster requires.



WARNING

If you do not choose to create firewall rules based on cluster events, you must create cluster-wide firewall rules.

Prerequisites

- You exported the variables that the Deployment Manager templates require to deploy your cluster.
- You created the networking and load balancing components in GCP that your cluster requires.

Procedure

1. Add a single firewall rule to allow the Google Cloud Engine health checks to access all of the services. This rule enables the ingress load balancers to determine the health status of their instances.

```
$ gcloud compute firewall-rules create --allow='tcp:30000-32767,udp:30000-32767' --
network="${CLUSTER_NETWORK}" --source-
ranges='130.211.0.0/22,35.191.0.0/16,209.85.152.0/22,209.85.204.0/22' --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress-hc --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

2. Add a single firewall rule to allow access to all cluster services:

- For an external cluster:

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --
network="${CLUSTER_NETWORK}" --source-ranges="0.0.0.0/0" --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

- For a private cluster:

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --
network="${CLUSTER_NETWORK}" --source-ranges=${NETWORK_CIDR} --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

Because this rule only allows traffic on TCP ports **80** and **443**, ensure that you add all the ports that your services use.

9.12.24. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.
- Install the **oc** CLI and log in.

Procedure

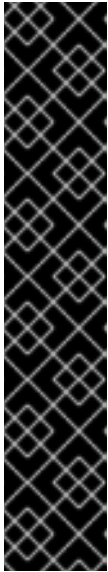
1. Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Observe the running state of your cluster.

- a. Run the following command to view the current cluster version and status:

```
$ oc get clusterversion
```

Example output

```
NAME      VERSION  AVAILABLE  PROGRESSING  SINCE  STATUS
version   False    True        24m          Working towards 4.5.4: 99% complete
```

- b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

```
$ oc get clusteroperators
```

Example output

```
NAME                               VERSION  AVAILABLE  PROGRESSING  DEGRADED  SINCE
authentication                      4.5.4   True        False        False     7m56s
cloud-credential                     4.5.4   True        False        False     31m
cluster-autoscaler                   4.5.4   True        False        False     16m
console                              4.5.4   True        False        False     10m
csi-snapshot-controller              4.5.4   True        False        False     16m
dns                                  4.5.4   True        False        False     22m
etcd                                  4.5.4   False       False        False     25s
image-registry                       4.5.4   True        False        False     16m
ingress                              4.5.4   True        False        False     16m
insights                             4.5.4   True        False        False     17m
kube-apiserver                       4.5.4   True        False        False     19m
kube-controller-manager              4.5.4   True        False        False     20m
kube-scheduler                       4.5.4   True        False        False     20m
```

kube-storage-version-migrator	4.5.4	True	False	False	16m
machine-api	4.5.4	True	False	False	22m
machine-config	4.5.4	True	False	False	22m
marketplace	4.5.4	True	False	False	16m
monitoring	4.5.4	True	False	False	10m
network	4.5.4	True	False	False	23m
node-tuning	4.5.4	True	False	False	23m
openshift-apiserver	4.5.4	True	False	False	17m
openshift-controller-manager	4.5.4	True	False	False	15m
openshift-samples	4.5.4	True	False	False	16m
operator-lifecycle-manager	4.5.4	True	False	False	22m
operator-lifecycle-manager-catalog	4.5.4	True	False	False	22m
operator-lifecycle-manager-packageserver	4.5.4	True	False	False	18m
service-ca	4.5.4	True	False	False	23m
service-catalog-apiserver	4.5.4	True	False	False	23m
service-catalog-controller-manager	4.5.4	True	False	False	23m
storage	4.5.4	True	False	False	17m

- c. Run the following command to view your cluster pods:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE                               NAME
READY STATUS  RESTARTS  AGE
kube-system          etcd-member-ip-10-0-3-111.us-east-
2.compute.internal  1/1      Running   0    35m
kube-system          etcd-member-ip-10-0-3-239.us-east-
2.compute.internal  1/1      Running   0    37m
kube-system          etcd-member-ip-10-0-3-24.us-east-
2.compute.internal  1/1      Running   0    35m
openshift-apiserver-operator            openshift-apiserver-operator-6d6674f4f4-
h7t2t          1/1      Running   1    37m
openshift-apiserver          apiserver-fm48r
1/1      Running   0    30m
openshift-apiserver          apiserver-fxkvv
1/1      Running   0    29m
openshift-apiserver          apiserver-q85nm
1/1      Running   0    29m
...
openshift-service-ca-operator            openshift-service-ca-operator-66ff6dc6cd-
9r257          1/1      Running   0    37m
openshift-service-ca          apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1      Running   0    35m
openshift-service-ca          configmap-cabundle-injector-8498544d7-
25qn6          1/1      Running   0    35m
openshift-service-ca          service-serving-cert-signer-6445fc9c6-wqdqn
1/1      Running   0    35m
openshift-service-catalog-apiserver-operator            openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w  1/1      Running   0    32m
openshift-service-catalog-controller-manager-operator            openshift-service-catalog-
controller-manager-operator-b78cr2lnm  1/1      Running   0    31m

```

When the current cluster version is **AVAILABLE**, the installation is complete.

9.12.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

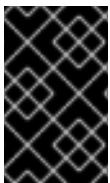
- See [About remote health monitoring](#) for more information about the Telemetry service

9.12.26. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

9.13. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on Google Cloud Platform (GCP) that uses infrastructure that you provide and an internal mirror of the installation release content.



IMPORTANT

While you can install an OpenShift Container Platform cluster by using mirrored installation release content, your cluster still requires internet access to use the GCP APIs.

The steps for performing a user-provided infrastructure install are outlined here. Several [Deployment Manager](#) templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.



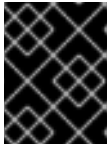
IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

9.13.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.

- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

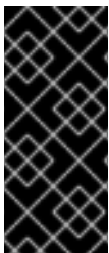
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to. While you might need to grant access to more sites, you must grant access to ***.googleapis.com** and **accounts.google.com**.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you can [manually create and maintain IAM credentials](#).

9.13.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

9.13.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

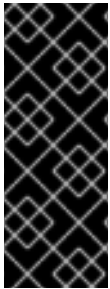
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

9.13.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

9.13.4. Configuring your GCP project

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

9.13.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See [Creating and Managing Projects](#) in the GCP documentation.



IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the **api-int.<cluster_name>.<base_domain>** URL; the Premium Tier is required for internal load balancing.

9.13.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. You may also enable optional API services which are not required for installation. See [Enabling services](#) in the GCP documentation.

Table 9.59. Required API services

API service	Console service name
Compute Engine API	compute.googleapis.com
Cloud Resource Manager API	cloudresourcemanager.googleapis.com
Google DNS API	dns.googleapis.com
IAM Service Account Credentials API	iamcredentials.googleapis.com
Identity and Access Management (IAM) API	iam.googleapis.com
Service Usage API	serviceusage.googleapis.com

Table 9.60. Optional API services

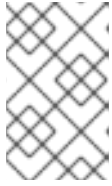
API service	Console service name
Google Cloud APIs	cloudapis.googleapis.com
Service Management API	servicemanagement.googleapis.com
Google Cloud Storage JSON API	storage-api.googleapis.com
Cloud Storage	storage-component.googleapis.com

9.13.4.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.

**NOTE**

If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](#).

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](#) in the GCP documentation.
Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](#) in the GCP documentation.
You typically have four name servers.
4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](#).
5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](#) in the GCP documentation.
6. If you use a subdomain, follow your company's procedures to add its delegation records to the parent domain. This process might include a request to your company's IT department or the division that controls the root domain and DNS services for your company.

9.13.4.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default [Quotas](#) do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

Table 9.61. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	6	1
Firewall rules	Networking	Global	11	1
Forwarding rules	Compute	Global	2	0
Health checks	Compute	Global	2	0
Images	Compute	Global	1	0
Networks	Networking	Global	1	0

Service	Component	Location	Total resources required	Resources removed after bootstrap
Routers	Networking	Global	1	0
Routes	Networking	Global	2	0
Subnetworks	Compute	Global	2	0
Target pools	Networking	Global	2	0

**NOTE**

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europa-north1**
- **europa-west2**
- **europa-west3**
- **europa-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

You can increase resource quotas from the [GCP console](#), but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

9.13.4.5. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See [Creating a service account](#) in the GCP documentation.
2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See [Granting roles to a service account for specific resources](#).

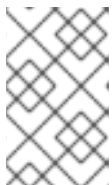


NOTE

While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. You can create the service account key in JSON format, or attach the service account to a GCP virtual machine. See [Creating service account keys](#) and [Creating and enabling service accounts for instances](#) in the GCP documentation.

You must have a service account key or a virtual machine with an attached service account to create the cluster.



NOTE

If you use a virtual machine with an attached service account to create your cluster, you must set **credentialsMode: Manual** in the **install-config.yaml** file before installation.

9.13.4.6. Required GCP roles

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. If the security policies for your organization require a more restrictive set of permissions, you can create a service account with the following permissions.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions.

If you deploy your cluster into an existing virtual private cloud (VPC), the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- IAM Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Required roles for using passthrough credentials mode

- Compute Load Balancer Admin
- IAM Role Viewer

Required roles for user-provisioned GCP infrastructure

- Deployment Manager Editor

The roles are applied to the service accounts that the control plane and compute machines use:

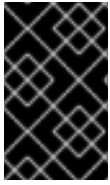
Table 9.62. GCP service account permissions

Account	Roles
Control Plane	roles/compute.instanceAdmin
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	roles/compute.viewer
	roles/storage.admin

9.13.4.7. Required GCP permissions for user-provisioned infrastructure

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform.

If the security policies for your organization require a more restrictive set of permissions, you can create [custom roles](#) with the necessary permissions. The following permissions are required for the user-provisioned infrastructure for creating and deleting the OpenShift Container Platform cluster.



IMPORTANT

If you configure the Cloud Credential Operator to operate in passthrough mode, you must use roles rather than granular permissions. For more information, see "Required roles for using passthrough credentials mode" in the "Required GCP roles" section.

Example 9.68. Required permissions for creating network resources

- `compute.addresses.create`
- `compute.addresses.createInternal`
- `compute.addresses.delete`
- `compute.addresses.get`
- `compute.addresses.list`
- `compute.addresses.use`
- `compute.addresses.useInternal`
- `compute.firewalls.create`
- `compute.firewalls.delete`
- `compute.firewalls.get`
- `compute.firewalls.list`
- `compute.forwardingRules.create`
- `compute.forwardingRules.get`
- `compute.forwardingRules.list`
- `compute.forwardingRules.setLabels`
- `compute.networks.create`
- `compute.networks.get`
- `compute.networks.list`
- `compute.networks.updatePolicy`
- `compute.routers.create`
- `compute.routers.get`
- `compute.routers.list`
- `compute.routers.update`

- `compute.routes.list`
- `compute.subnetworks.create`
- `compute.subnetworks.get`
- `compute.subnetworks.list`
- `compute.subnetworks.use`
- `compute.subnetworks.useExternallp`

Example 9.69. Required permissions for creating load balancer resources

- `compute.regionBackendServices.create`
- `compute.regionBackendServices.get`
- `compute.regionBackendServices.list`
- `compute.regionBackendServices.update`
- `compute.regionBackendServices.use`
- `compute.targetPools.addInstance`
- `compute.targetPools.create`
- `compute.targetPools.get`
- `compute.targetPools.list`
- `compute.targetPools.removeInstance`
- `compute.targetPools.use`

Example 9.70. Required permissions for creating DNS resources

- `dns.changes.create`
- `dns.changes.get`
- `dns.managedZones.create`
- `dns.managedZones.get`
- `dns.managedZones.list`
- `dns.networks.bindPrivateDNSZone`
- `dns.resourceRecordSets.create`
- `dns.resourceRecordSets.list`
- `dns.resourceRecordSets.update`

Example 9.71. Required permissions for creating Service Account resources

- `iam.serviceAccountKeys.create`
- `iam.serviceAccountKeys.delete`
- `iam.serviceAccountKeys.get`
- `iam.serviceAccountKeys.list`
- `iam.serviceAccounts.actAs`
- `iam.serviceAccounts.create`
- `iam.serviceAccounts.delete`
- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.get`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.72. Required permissions for creating compute resources

- `compute.disks.create`
- `compute.disks.get`
- `compute.disks.list`
- `compute.instanceGroups.create`
- `compute.instanceGroups.delete`
- `compute.instanceGroups.get`
- `compute.instanceGroups.list`
- `compute.instanceGroups.update`
- `compute.instanceGroups.use`
- `compute.instances.create`
- `compute.instances.delete`
- `compute.instances.get`
- `compute.instances.list`
- `compute.instances.setLabels`

- **compute.instances.setMetadata**
- **compute.instances.setServiceAccount**
- **compute.instances.setTags**
- **compute.instances.use**
- **compute.machineTypes.get**
- **compute.machineTypes.list**

Example 9.73. Required for creating storage resources

- **storage.buckets.create**
- **storage.buckets.delete**
- **storage.buckets.get**
- **storage.buckets.list**
- **storage.objects.create**
- **storage.objects.delete**
- **storage.objects.get**
- **storage.objects.list**

Example 9.74. Required permissions for creating health check resources

- **compute.healthChecks.create**
- **compute.healthChecks.get**
- **compute.healthChecks.list**
- **compute.healthChecks.useReadOnly**
- **compute.httpHealthChecks.create**
- **compute.httpHealthChecks.get**
- **compute.httpHealthChecks.list**
- **compute.httpHealthChecks.useReadOnly**

Example 9.75. Required permissions to get GCP zone and region related information

- **compute.globalOperations.get**
- **compute.regionOperations.get**

- `compute.regions.list`
- `compute.zoneOperations.get`
- `compute.zones.get`
- `compute.zones.list`

Example 9.76. Required permissions for checking services and quotas

- `monitoring.timeSeries.list`
- `serviceusage.quotas.get`
- `serviceusage.services.list`

Example 9.77. Required IAM permissions for installation

- `iam.roles.get`

Example 9.78. Required Images permissions for installation

- `compute.images.create`
- `compute.images.delete`
- `compute.images.get`
- `compute.images.list`

Example 9.79. Optional permission for running gather bootstrap

- `compute.instances.getSerialPortOutput`

Example 9.80. Required permissions for deleting network resources

- `compute.addresses.delete`
- `compute.addresses.deleteInternal`
- `compute.addresses.list`
- `compute.firewalls.delete`
- `compute.firewalls.list`
- `compute.forwardingRules.delete`
- `compute.forwardingRules.list`

- `compute.networks.delete`
- `compute.networks.list`
- `compute.networks.updatePolicy`
- `compute.routers.delete`
- `compute.routers.list`
- `compute.routes.list`
- `compute.subnetworks.delete`
- `compute.subnetworks.list`

Example 9.81. Required permissions for deleting load balancer resources

- `compute.regionBackendServices.delete`
- `compute.regionBackendServices.list`
- `compute.targetPools.delete`
- `compute.targetPools.list`

Example 9.82. Required permissions for deleting DNS resources

- `dns.changes.create`
- `dns.managedZones.delete`
- `dns.managedZones.get`
- `dns.managedZones.list`
- `dns.resourceRecordSets.delete`
- `dns.resourceRecordSets.list`

Example 9.83. Required permissions for deleting Service Account resources

- `iam.serviceAccounts.delete`
- `iam.serviceAccounts.get`
- `iam.serviceAccounts.list`
- `resourcemanager.projects.getIamPolicy`
- `resourcemanager.projects.setIamPolicy`

Example 9.84. Required permissions for deleting compute resources

- **compute.disks.delete**
- **compute.disks.list**
- **compute.instanceGroups.delete**
- **compute.instanceGroups.list**
- **compute.instances.delete**
- **compute.instances.list**
- **compute.instances.stop**
- **compute.machineTypes.list**

Example 9.85. Required for deleting storage resources

- **storage.buckets.delete**
- **storage.buckets.getIamPolicy**
- **storage.buckets.list**
- **storage.objects.delete**
- **storage.objects.list**

Example 9.86. Required permissions for deleting health check resources

- **compute.healthChecks.delete**
- **compute.healthChecks.list**
- **compute.httpHealthChecks.delete**
- **compute.httpHealthChecks.list**

Example 9.87. Required Images permissions for deletion

- **compute.images.delete**
- **compute.images.list**

Example 9.88. Required permissions to get Region related information

- **compute.regions.get**

Example 9.89. Required Deployment Manager permissions

- **deploymentmanager.deployments.create**
- **deploymentmanager.deployments.delete**
- **deploymentmanager.deployments.get**
- **deploymentmanager.deployments.list**
- **deploymentmanager.manifests.get**
- **deploymentmanager.operations.get**
- **deploymentmanager.resources.list**

Additional resources

- [Optimizing storage](#)

9.13.4.8. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **australia-southeast1** (Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **eu-central2** (Warsaw, Poland)
- **eu-north1** (Hamina, Finland)
- **eu-southwest1** (Madrid, Spain)
- **eu-west1** (St. Ghislain, Belgium)
- **eu-west2** (London, England, UK)

- **europa-west3** (Frankfurt, Germany)
- **europa-west4** (Eemshaven, Netherlands)
- **europa-west6** (Zürich, Switzerland)
- **europa-west8** (Milan, Italy)
- **europa-west9** (Paris, France)
- **europa-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)



NOTE

To determine which machine type instances are available by region and zone, see the Google [documentation](#).

9.13.4.9. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

Prerequisites

- You created a project to host your cluster.
- You created a service account and granted it the required permissions.

Procedure

1. Install the following binaries in **\$PATH**:

- **gcloud**
- **gsutil**

See [Install the latest Cloud SDK version](#) in the GCP documentation.

2. Authenticate using the **gcloud** tool with your configured service account.

See [Authorizing with a service account](#) in the GCP documentation.

9.13.5. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

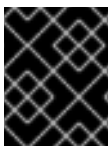
This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

9.13.5.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 9.63. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

9.13.5.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 9.64. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

9.13.5.3. Tested instance types for GCP

The following Google Cloud Platform instance types have been tested with OpenShift Container Platform.

Example 9.90. Machine series

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**

- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.13.5.4. Using custom machine types

Using a custom machine type to install a OpenShift Container Platform cluster is supported.

Consider the following when using a custom machine type:

- Similar to predefined instance types, custom machine types must meet the minimum resource requirements for control plane and compute machines. For more information, see "Minimum resource requirements for cluster installation".
- The name of the custom machine type must adhere to the following syntax:
custom-<number_of_cpus>-<amount_of_memory_in_mb>

For example, **custom-6-20480**.

9.13.6. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

9.13.6.1. Optional: Creating a separate **/var** partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
```

```

name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> ❶
      partitions:
        - label: var
          start_mib: <partition_start_offset> ❷
          size_mib: <partition_size> ❸
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] ❹
          with_mount_unit: true

```

- ❶ The storage device name of the disk that you want to partition.
- ❷ When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- ❸ The size of the data partition in mebibytes.
- ❹ The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

9.13.6.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **gcp** as the platform to target.
- iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

- iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.
 - v. Select the region to deploy the cluster to.
 - vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - vii. Enter a descriptive name for your cluster.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.

- a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000":{"auth": "<credentials>","email": "you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Define the network and subnets for the VPC to install the cluster in under the parent **platform.gcp** field:

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

For **platform.gcp.network**, specify the name for the existing Google VPC. For **platform.gcp.controlPlaneSubnet** and **platform.gcp.computeSubnet**, specify the existing subnets to deploy the control plane machines and compute machines, respectively.

- d. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

3. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.13.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
```

```
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

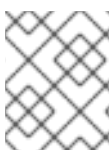
**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

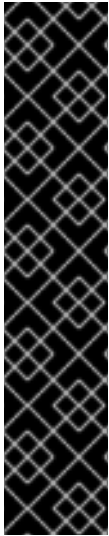
**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

9.13.6.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Optional: If you do not want the cluster to provision compute machines, remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
5. Optional: If you do not want [the Ingress Operator](#) to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

```

apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: ❶
    id: mycluster-100419-private-zone
  publicZone: ❷
    id: example.openshift.com
status: {}

```

- ❶ ❷ Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> ❶
```

- ❶ For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:

```

.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign

```

Additional resources

- [Optional: Adding the ingress DNS records](#)

9.13.7. Exporting common variables

9.13.7.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

9.13.7.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).



NOTE

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

- Generate the Ignition config files for your cluster.
- Install the **jq** package.

Procedure

1. Export the following common variables to be used by the provided Deployment Manager templates:

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/17'
$ export WORKER_SUBNET_CIDR='10.0.128.0/17'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

9.13.8. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the **Deployment Manager template for the VPC** section of this topic and save it as **01_vpc.py** on your computer. This template describes the VPC that your cluster requires.
2. Create a **01_vpc.yaml** resource definition file:

```
$ cat <<EOF >01_vpc.yaml
imports:
```

```

- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' ❶
    region: '${REGION}' ❷
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' ❸
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' ❹
EOF

```

- ❶ **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- ❷ **region** is the region to deploy the cluster into, for example **us-central1**.
- ❸ **master_subnet_cidr** is the CIDR for the master subnet, for example **10.0.0.0/17**.
- ❹ **worker_subnet_cidr** is the CIDR for the worker subnet, for example **10.0.128.0/17**.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

9.13.8.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

Example 9.91. 01_vpc.py Deployment Manager template

```

def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',

```

```

        'ipCidrRange': context.properties['worker_subnet_cidr']
    }
}, {
    'name': context.properties['infra_id'] + '-router',
    'type': 'compute.v1.router',
    'properties': {
        'region': context.properties['region'],
        'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
        'nats': [{
            'name': context.properties['infra_id'] + '-nat-master',
            'natIpAllocateOption': 'AUTO_ONLY',
            'minPortsPerVm': 7168,
            'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
            'subnetworks': [{
                'name': '${ref.' + context.properties['infra_id'] + '-master-subnet.selfLink}',
                'sourceIpRangesToNat': ['ALL_IP_RANGES']
            }]
        }]
    }, {
        'name': context.properties['infra_id'] + '-nat-worker',
        'natIpAllocateOption': 'AUTO_ONLY',
        'minPortsPerVm': 512,
        'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
        'subnetworks': [{
            'name': '${ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink}',
            'sourceIpRangesToNat': ['ALL_IP_RANGES']
        }]
    }
}
}
}}
return {'resources': resources}

```

9.13.9. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

9.13.9.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

9.13.9.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

Table 9.65. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 9.66. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 9.67. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

9.13.10. Creating load balancers in GCP

You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the internal load balancer** section of this topic and save it as **02_lb_int.py** on your computer. This template describes the internal load balancing objects that your cluster requires.
2. For an external cluster, also copy the template from the **Deployment Manager template for the external load balancer** section of this topic and save it as **02_lb_ext.py** on your computer. This template describes the external load balancing objects that your cluster requires.
3. Export the variables that the deployment template uses:

- a. Export the cluster network location:

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
```

- b. Export the control plane subnet location:

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

- c. Export the three zones that the cluster uses:

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9`)
```


4. Create a **02_infra.yaml** resource definition file:

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py 1
resources:
- name: cluster-lb-ext 2
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' 3
    region: '${REGION}' 4
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' 5
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: 6
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

- 1 2 Required only when deploying an external cluster.
- 3 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 4 **region** is the region to deploy the cluster into, for example **us-central1**.
- 5 **control_subnet** is the URI to the control subnet.
- 6 **zones** are the zones to deploy the control plane instances into, like **us-east1-b**, **us-east1-c**, and **us-east1-d**.

5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. For an external cluster, also export the cluster public IP address:

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

9.13.10.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

Example 9.92. 02_lb_ext.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
        # probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$ (ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$ (ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$ (ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }
    ]

    return {'resources': resources}
```

9.13.10.2. Deployment Manager template for the internal load balancer

You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

Example 9.93. 02_lb_int.py Deployment Manager template

```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
```

```

backends.append({
  'group': '${ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink}'
})

resources = [{
  'name': context.properties['infra_id'] + '-cluster-ip',
  'type': 'compute.v1.address',
  'properties': {
    'addressType': 'INTERNAL',
    'region': context.properties['region'],
    'subnetwork': context.properties['control_subnet']
  }
}, {
  # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
  # probe for kube-apiserver
  'name': context.properties['infra_id'] + '-api-internal-health-check',
  'type': 'compute.v1.healthCheck',
  'properties': {
    'httpsHealthCheck': {
      'port': 6443,
      'requestPath': '/readyz'
    },
    'type': "HTTPS"
  }
}, {
  'name': context.properties['infra_id'] + '-api-internal-backend-service',
  'type': 'compute.v1.regionBackendService',
  'properties': {
    'backends': backends,
    'healthChecks': ['${ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink}'],
    'loadBalancingScheme': 'INTERNAL',
    'region': context.properties['region'],
    'protocol': 'TCP',
    'timeoutSec': 120
  }
}, {
  'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
  'type': 'compute.v1.forwardingRule',
  'properties': {
    'backendService': '${ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink}',
    'IPAddress': '${ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink}',
    'loadBalancingScheme': 'INTERNAL',
    'ports': ['6443','22623'],
    'region': context.properties['region'],
    'subnetwork': context.properties['control_subnet']
  }
}
]]

for zone in context.properties['zones']:
  resources.append({
    'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
      'namedPorts': [

```

```

    {
      'name': 'ignition',
      'port': 22623
    }, {
      'name': 'https',
      'port': 6443
    }
  ],
  'network': context.properties['cluster_network'],
  'zone': zone
}
})

return {'resources': resources}

```

You will need this template in addition to the **02_lb_ext.py** template when you create an external cluster.

9.13.11. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for the private DNS** section of this topic and save it as **02_dns.py** on your computer. This template describes the private DNS objects that your cluster requires.
2. Create a **02_dns.yaml** resource definition file:

```

$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py

```

```

properties:
  infra_id: '${INFRA_ID}' 1
  cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' 2
  cluster_network: '${CLUSTER_NETWORK}' 3
EOF

```

- 1** `infra_id` is the `INFRA_ID` infrastructure name from the extraction step.
- 2** `cluster_domain` is the domain for the cluster, for example `openshift.example.com`.
- 3** `cluster_network` is the `selfLink` URL to the cluster network.

3. Create the deployment by using the `gcloud` CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

- a. Add the internal DNS entries:

```

$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone

```

- b. For an external cluster, also add the external DNS entries:

```

$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}

```

9.13.11.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

Example 9.94. `02_dns.py` Deployment Manager template

```

def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {

```

```

'description': "",
'dnsName': context.properties['cluster_domain'] + '.',
'visibility': 'private',
'privateVisibilityConfig': {
  'networks': [{
    'networkUrl': context.properties['cluster_network']
  }]
}
}
}]

return {'resources': resources}

```

9.13.12. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for firewall rules** section of this topic and save it as **03_firewall.py** on your computer. This template describes the security groups that your cluster requires.
2. Create a **03_firewall.yaml** resource definition file:

```

$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' 1
    infra_id: '${INFRA_ID}' 2

```

```

cluster_network: '${CLUSTER_NETWORK}' 3
network_cidr: '${NETWORK_CIDR}' 4
EOF

```

- 1 **allowed_external_cidr** is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to **\${NETWORK_CIDR}**.
- 2 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 3 **cluster_network** is the **selfLink** URL to the cluster network.
- 4 **network_cidr** is the CIDR of the VPC network, for example **10.0.0.0/16**.

3. Create the deployment by using the **gcloud** CLI:

```

$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml

```

9.13.12.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

Example 9.95. 03_firewall.py Deployment Manager template

```

def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    ], {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',

```

```

'properties': {
  'network': context.properties[cluster_network],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['6080', '6443', '22624']
  }],
  'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
  'targetTags': [context.properties[infra_id] + '-master']
}
}, {
  'name': context.properties[infra_id] + '-etcd',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties[cluster_network],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['2379-2380']
    }],
    'sourceTags': [context.properties[infra_id] + '-master'],
    'targetTags': [context.properties[infra_id] + '-master']
  }
}, {
  'name': context.properties[infra_id] + '-control-plane',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties[cluster_network],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['10257']
    }],
    {
      'IPProtocol': 'tcp',
      'ports': ['10259']
    },
    {
      'IPProtocol': 'tcp',
      'ports': ['22623']
    }
  ],
  'sourceTags': [
    context.properties[infra_id] + '-master',
    context.properties[infra_id] + '-worker'
  ],
  'targetTags': [context.properties[infra_id] + '-master']
}
}, {
  'name': context.properties[infra_id] + '-internal-network',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties[cluster_network],
    'allowed': [{
      'IPProtocol': 'icmp'
    }],
    {
      'IPProtocol': 'tcp',
      'ports': ['22']
    }
  ],
  'sourceRanges': [context.properties[network_cidr]],
  'targetTags': [
    context.properties[infra_id] + '-master',

```



```

        context.properties['infra_id'] + '-worker'
    ]
  }
}, {
  'name': context.properties['infra_id'] + '-internal-cluster',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'udp',
      'ports': ['4789', '6081']
    }, {
      'IPProtocol': 'udp',
      'ports': ['500', '4500']
    }, {
      'IPProtocol': 'esp',
    }, {
      'IPProtocol': 'tcp',
      'ports': ['9000-9999']
    }, {
      'IPProtocol': 'udp',
      'ports': ['9000-9999']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['10250']
    }, {
      'IPProtocol': 'tcp',
      'ports': ['30000-32767']
    }, {
      'IPProtocol': 'udp',
      'ports': ['30000-32767']
    }
  ],
  'sourceTags': [
    context.properties['infra_id'] + '-master',
    context.properties['infra_id'] + '-worker'
  ],
  'targetTags': [
    context.properties['infra_id'] + '-master',
    context.properties['infra_id'] + '-worker'
  ]
}
]]

return {'resources': resources}

```

9.13.13. Creating IAM roles in GCP

You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the **Deployment Manager template for IAM roles** section of this topic and save it as **03_iam.py** on your computer. This template describes the IAM roles that your cluster requires.
2. Create a **03_iam.yaml** resource definition file:

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' 1
EOF
```

- 1** **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. Export the variable for the master service account:

```
$ export MASTER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

5. Export the variable for the worker service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

6. Export the variable for the subnet that hosts the compute machines:

```
$ export COMPUTE_SUBNET=$(gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink')
```

- The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

- Create a service account key and store it locally for later use:

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.13.13.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

Example 9.96. 03_iam.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }
    ]

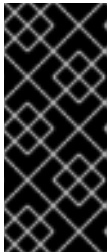
    return {'resources': resources}
```

9.13.14. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.

Procedure

1. Obtain the RHCOS image from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The file name contains the OpenShift Container Platform version number in the format **rhcos-
<version>-<arch>-gcp.<arch>.tar.gz**.

2. Create the Google storage bucket:

```
$ gsutil mb gs://<bucket_name>
```

3. Upload the RHCOS image to the Google storage bucket:

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz  
gs://<bucket_name>
```

4. Export the uploaded RHCOS image location as a variable:

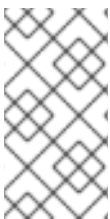
```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. Create the cluster image:

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \  
--source-uri="${IMAGE_SOURCE}"
```

9.13.15. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.

- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Ensure pyOpenSSL is installed.

Procedure

1. Copy the template from the **Deployment Manager template for the bootstrap machine** section of this topic and save it as **04_bootstrap.py** on your computer. This template describes the bootstrap machine that your cluster requires.
2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --
format json | jq -r .selfLink`)
```

3. Create a bucket and upload the **bootstrap.ign** file:

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-
bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. Create a **04_bootstrap.yaml** resource definition file:

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    zone: '${ZONE_0}' 3

    cluster_network: '${CLUSTER_NETWORK}' 4
    control_subnet: '${CONTROL_SUBNET}' 5
    image: '${CLUSTER_IMAGE}' 6
    machine_type: 'n1-standard-4' 7
    root_volume_size: '128' 8
```

```
bootstrap_ign: '${BOOTSTRAP_IGN}' 9
EOF
```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **region** is the region to deploy the cluster into, for example **us-central1**.
- 3 **zone** is the zone to deploy the bootstrap instance into, for example **us-central1-b**.
- 4 **cluster_network** is the **selfLink** URL to the cluster network.
- 5 **control_subnet** is the **selfLink** URL to the control subnet.
- 6 **image** is the **selfLink** URL to the RHCOS image.
- 7 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 8 **root_volume_size** is the boot disk size for the bootstrap machine.
- 9 **bootstrap_ign** is the URL output when creating a signed URL.

6. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the bootstrap machine manually.
 - a. Add the bootstrap instance to the internal load balancer instance group:

```
$ gcloud compute instance-groups unmanaged add-instances \
  ${INFRA_ID}-bootstrap-ig --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

- b. Add the bootstrap instance group to the internal load balancer backend service:

```
$ gcloud compute backend-services add-backend \
  ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
  group=${INFRA_ID}-bootstrap-ig --instance-group-zone=${ZONE_0}
```

9.13.15.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 9.97. 04_bootstrap.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {
```

```

        'region': context.properties['region']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"},"version":"3.2.0"}}}',
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet'],
            'accessConfigs': [{
                'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
            }]
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-bootstrap'
            ]
        },
        'zone': context.properties['zone']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}
]]

return {'resources': resources}

```

9.13.16. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.



NOTE

If you do not use the provided Deployment Manager template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the **Deployment Manager template for control plane machines** section of this topic and save it as **05_control_plane.py** on your computer. This template describes the control plane machines that your cluster requires.
2. Export the following variable required by the resource definition:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. Create a **05_control_plane.yaml** resource definition file:

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' 1
    zones: 2
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
```



```

image: '${CLUSTER_IMAGE}' 4
machine_type: 'n1-standard-4' 5
root_volume_size: '128'
service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

ignition: '${MASTER_IGNITION}' 7
EOF

```

- 1 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 2 **zones** are the zones to deploy the control plane instances into, for example **us-central1-a**, **us-central1-b**, and **us-central1-c**.
- 3 **control_subnet** is the **selfLink** URL to the control subnet.
- 4 **image** is the **selfLink** URL to the RHCOS image.
- 5 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 6 **service_account_email** is the email address for the master service account that you created.
- 7 **ignition** is the contents of the **master.ign** file.

4. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

- Run the following commands to add the control plane machines to the appropriate instance groups:

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

- For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.13.16.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 9.98. 05_control_plane.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    ], {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
```

```

        'diskSizeGb': context.properties['root_volume_size'],
        'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
        'sourceImage': context.properties['image']
    }
  }],
  'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
  'metadata': {
    'items': [{
      'key': 'user-data',
      'value': context.properties['ignition']
    }]
  },
  'networkInterfaces': [{
    'subnetwork': context.properties['control_subnet']
  }],
  'serviceAccounts': [{
    'email': context.properties['service_account_email'],
    'scopes': ['https://www.googleapis.com/auth/cloud-platform']
  }],
  'tags': {
    'items': [
      context.properties['infra_id'] + '-master',
    ]
  },
  'zone': context.properties['zones'][1]
}
}, {
  'name': context.properties['infra_id'] + '-master-2',
  'type': 'compute.v1.instance',
  'properties': {
    'disks': [{
      'autoDelete': True,
      'boot': True,
      'initializeParams': {
        'diskSizeGb': context.properties['root_volume_size'],
        'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
        'sourceImage': context.properties['image']
      }
    }]
  },
  'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
  'metadata': {
    'items': [{
      'key': 'user-data',
      'value': context.properties['ignition']
    }]
  },
  'networkInterfaces': [{
    'subnetwork': context.properties['control_subnet']
  }],
  'serviceAccounts': [{
    'email': context.properties['service_account_email'],
    'scopes': ['https://www.googleapis.com/auth/cloud-platform']
  }],
  'tags': {

```

```

        'items': [
            context.properties['infra_id'] + '-master',
        ]
    },
    'zone': context.properties['zones'][2]
}
}}

return {'resources': resources}

```

9.13.17. Wait for bootstrap completion and remove bootstrap resources in GCP

After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
--log-level info 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```

```
$ gsutil rm gs://{INFRA_ID}-bootstrap-ignition/bootstrap.ign
```

```
$ gsutil rb gs://{INFRA_ID}-bootstrap-ignition
```

```
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

9.13.18. Creating additional worker machines in GCP

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type **06_worker.py** in the file.



NOTE

If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the **Deployment Manager template for worker machines** section of this topic and save it as **06_worker.py** on your computer. This template describes the worker machines that your cluster requires.
2. Export the variables that the resource definition uses.
 - a. Export the subnet that hosts the compute machines:

```
$ export COMPUTE_SUBNET=`gcloud compute networks subnets describe  
${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink`
```

- b. Export the email address for your service account:

```
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '[0].email')
```

- c. Export the location of the compute machine Ignition config file:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. Create a **06_worker.yaml** resource definition file:

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' 1
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' 2
    zone: '${ZONE_0}' 3
    compute_subnet: '${COMPUTE_SUBNET}' 4
    image: '${CLUSTER_IMAGE}' 5
    machine_type: 'n1-standard-4' 6
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 7
    ignition: '${WORKER_IGNITION}' 8
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' 9
    zone: '${ZONE_1}' 10
    compute_subnet: '${COMPUTE_SUBNET}' 11
    image: '${CLUSTER_IMAGE}' 12
    machine_type: 'n1-standard-4' 13
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' 14
    ignition: '${WORKER_IGNITION}' 15
EOF
```

- 1 **name** is the name of the worker machine, for example **worker-0**.
- 2 9 **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- 3 10 **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.
- 4 11 **compute_subnet** is the **selfLink** URL to the compute subnet.
- 5 12 **image** is the **selfLink** URL to the RHCOS image. ¹
- 6 13 **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- 7 14 **service_account_email** is the email address for the worker service account that you created.

8 15 **ignition** is the contents of the **worker.ign** file.

4. Optional: If you want to launch additional instances, include additional resources of type **06_worker.py** in your **06_worker.yaml** resource definition file.
5. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

1. To use a GCP Marketplace image, specify the offer to use:
 - OpenShift Container Platform:
<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-48-x86-64-202210040145>
 - OpenShift Platform Plus: **<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-48-x86-64-202206140145>**
 - OpenShift Kubernetes Engine:
<https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-48-x86-64-202206140145>

9.13.18.1. Deployment Manager template for worker machines

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 9.99. 06_worker.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
        }],
```

```

    'serviceAccounts': [{
      'email': context.properties['service_account_email'],
      'scopes': ['https://www.googleapis.com/auth/cloud-platform']
    }],
    'tags': {
      'items': [
        context.properties['infra_id'] + '-worker',
      ]
    },
    'zone': context.properties['zone']
  }
}]

return {'resources': resources}

```

9.13.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

9.13.20. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

9.13.21. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 63m  v1.25.0
master-1  Ready   master 63m  v1.25.0
master-2  Ready   master 64m  v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE  REQUESTOR      CONDITION
```

```
csr-8b2br 15m system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

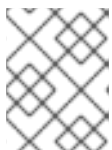
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   73m   v1.25.0
master-1  Ready   master   73m   v1.25.0
master-2  Ready   master   74m   v1.25.0
worker-0  Ready   worker   11m   v1.25.0
worker-1  Ready   worker   11m   v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

9.13.22. Optional: Adding the ingress DNS records

If you removed the DNS zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}**. or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- Configure a GCP account.
- Remove the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

Procedure

1. Wait for the Ingress router to create a load balancer and populate the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
router-default LoadBalancer  172.30.18.154  35.233.157.184 80:32288/TCP,443:31215/TCP 98
```

2. Add the A record to your zones:

- To use A records:
 - i. Export the variable for the router IP address:

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

- ii. Add the A record to the private zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

- iii. For an external cluster, also add the A record to the public zones:

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

- To add explicit domains instead of using a wildcard, create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

9.13.23. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.
- Install the **oc** CLI and log in.

Procedure

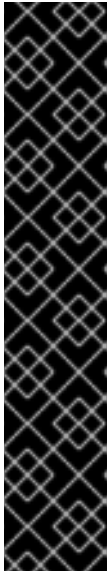
1. Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Observe the running state of your cluster.

- a. Run the following command to view the current cluster version and status:

```
$ oc get clusterversion
```

Example output

```
NAME      VERSION AVAILABLE PROGRESSING SINCE STATUS
version   False      True      24m      Working towards 4.5.4: 99% complete
```

- b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

```
$ oc get clusteroperators
```

Example output

```
NAME                               VERSION AVAILABLE PROGRESSING DEGRADED
SINCE
authentication                      4.5.4 True      False      False      7m56s
cloud-credential                    4.5.4 True      False      False      31m
cluster-autoscaler                  4.5.4 True      False      False      16m
console                             4.5.4 True      False      False      10m
csi-snapshot-controller              4.5.4 True      False      False      16m
dns                                  4.5.4 True      False      False      22m
etcd                                  4.5.4 False     False      False      25s
image-registry                      4.5.4 True      False      False      16m
ingress                              4.5.4 True      False      False      16m
insights                             4.5.4 True      False      False      17m
kube-apiserver                      4.5.4 True      False      False      19m
kube-controller-manager              4.5.4 True      False      False      20m
kube-scheduler                      4.5.4 True      False      False      20m
kube-storage-version-migrator        4.5.4 True      False      False      16m
machine-api                         4.5.4 True      False      False      22m
machine-config                      4.5.4 True      False      False      22m
marketplace                         4.5.4 True      False      False      16m
```

monitoring	4.5.4	True	False	False	10m
network	4.5.4	True	False	False	23m
node-tuning	4.5.4	True	False	False	23m
openshift-apiserver	4.5.4	True	False	False	17m
openshift-controller-manager	4.5.4	True	False	False	15m
openshift-samples	4.5.4	True	False	False	16m
operator-lifecycle-manager	4.5.4	True	False	False	22m
operator-lifecycle-manager-catalog	4.5.4	True	False	False	22m
operator-lifecycle-manager-packageserver	4.5.4	True	False	False	18m
service-ca	4.5.4	True	False	False	23m
service-catalog-apiserver	4.5.4	True	False	False	23m
service-catalog-controller-manager	4.5.4	True	False	False	23m
storage	4.5.4	True	False	False	17m

- c. Run the following command to view your cluster pods:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE                               NAME
READY  STATUS  RESTARTS  AGE
kube-system                               etcd-member-ip-10-0-3-111.us-east-
2.compute.internal                       1/1    Running  0    35m
kube-system                               etcd-member-ip-10-0-3-239.us-east-
2.compute.internal                       1/1    Running  0    37m
kube-system                               etcd-member-ip-10-0-3-24.us-east-
2.compute.internal                       1/1    Running  0    35m
openshift-apiserver-operator              openshift-apiserver-operator-6d6674f4f-
h7t2t                                    1/1    Running  1    37m
openshift-apiserver                       apiserver-fm48r
1/1    Running  0    30m
openshift-apiserver                       apiserver-fxkvv
1/1    Running  0    29m
openshift-apiserver                       apiserver-q85nm
1/1    Running  0    29m
...
openshift-service-ca-operator              openshift-service-ca-operator-66ff6dc6cd-
9r257                                    1/1    Running  0    37m
openshift-service-ca                      apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1    Running  0    35m
openshift-service-ca                      configmap-cabundle-injector-8498544d7-
25qn6                                    1/1    Running  0    35m
openshift-service-ca                      service-serving-cert-signer-6445fc9c6-wqdqn
1/1    Running  0    35m
openshift-service-catalog-apiserver-operator  openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w                1/1    Running  0    32m
openshift-service-catalog-controller-manager-operator  openshift-service-catalog-
controller-manager-operator-b78cr2lnm    1/1    Running  0    31m

```

When the current cluster version is **AVAILABLE**, the installation is complete.

9.13.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.13.25. Next steps

- [Customize your cluster](#).
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, see [Registering your disconnected cluster](#)

9.14. UNINSTALLING A CLUSTER ON GCP

You can remove a cluster that you deployed to Google Cloud Platform (GCP).

9.14.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access. For example, some Google Cloud resources require [IAM permissions](#) in shared VPC host projects, or there might be unused [health checks that must be deleted](#).

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

9.14.2. Deleting GCP resources with the Cloud Credential Operator utility

To clean up resources after uninstalling an OpenShift Container Platform cluster with the Cloud Credential Operator (CCO) in manual mode with GCP Workload Identity, you can use the CCO utility (**ccoctl**) to remove the GCP resources that **ccoctl** created during installation.

Prerequisites

- Extract and prepare the **ccoctl** binary.
- Install an OpenShift Container Platform cluster with the CCO in manual mode with GCP Workload Identity.

Procedure

1. Obtain the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk 'release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract --credentials-requests \
--cloud=gcp \
--to=<path_to_directory_with_list_of_credentials_requests>/credrequests 1
$RELEASE_IMAGE
```

- 1** **credrequests** is the directory where the list of **CredentialsRequest** objects is stored. This command creates the directory if it does not exist.

3. Delete the GCP resources that **ccoctl** created:

```
$ ccoctl gcp delete \  
  --name=<name> \ 1  
  --project=<gcp_project_id> \ 2  
  --credentials-requests-dir=  
<path_to_directory_with_list_of_credentials_requests>/credrequests
```

1 **<name>** matches the name that was originally used to create and tag the cloud resources.

2 **<gcp_project_id>** is the GCP project ID in which to delete cloud resources.

Verification

- To verify that the resources are deleted, query GCP. For more information, refer to GCP documentation.

CHAPTER 10. INSTALLING ON IBM CLOUD VPC

10.1. PREPARING TO INSTALL ON IBM CLOUD VPC

The installation workflows documented in this section are for IBM Cloud VPC infrastructure environments. IBM Cloud Classic is not supported at this time. For more information about the difference between Classic and VPC infrastructures, see the IBM [documentation](#).

10.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

10.1.2. Requirements for installing OpenShift Container Platform on IBM Cloud VPC

Before installing OpenShift Container Platform on IBM Cloud VPC, you must create a service account and configure an IBM Cloud account. See [Configuring an IBM Cloud account](#) for details about creating an account, enabling API services, configuring DNS, IBM Cloud account limits, and supported IBM Cloud VPC regions.

You must manually manage your cloud credentials when installing a cluster to IBM Cloud VPC. Do this by configuring the Cloud Credential Operator (CCO) for manual mode before you install the cluster. For more information, see [Configuring IAM for IBM Cloud VPC](#).

10.1.3. Choosing a method to install OpenShift Container Platform on IBM Cloud VPC

You can install OpenShift Container Platform on IBM Cloud VPC using installer-provisioned infrastructure. This process involves using an installation program to provision the underlying infrastructure for your cluster. Installing OpenShift Container Platform on IBM Cloud VPC using user-provisioned infrastructure is not supported at this time.

See [Installation process](#) for more information about installer-provisioned installation processes.

10.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on IBM Cloud VPC infrastructure that is provisioned by the OpenShift Container Platform installation program by using one of the following methods:

- **Installing a customized cluster on IBM Cloud VPC** You can install a customized cluster on IBM Cloud VPC infrastructure that the installation program provisions. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).
- **Installing a cluster on IBM Cloud VPC with network customizations** You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- **Installing a cluster on IBM Cloud VPC into an existing VPC** You can install OpenShift Container Platform on an existing IBM Virtual Private Cloud (VPC). You can use this installation method if you have constraints set by the guidelines of your company, such as limits when

creating new accounts or infrastructure.

- [Installing a private cluster on an existing VPC](#) You can install a private cluster on an existing Virtual Private Cloud (VPC). You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.

10.1.4. Next steps

- [Configuring an IBM Cloud account](#)

10.2. CONFIGURING AN IBM CLOUD ACCOUNT

Before you can install OpenShift Container Platform, you must configure an IBM Cloud account.

10.2.1. Prerequisites

- You have an IBM Cloud account with a subscription. You cannot install OpenShift Container Platform on a free or trial IBM Cloud account.

10.2.2. Quotas and limits on IBM Cloud VPC

The OpenShift Container Platform cluster uses a number of IBM Cloud VPC components, and the default quotas and limits affect your ability to install OpenShift Container Platform clusters. If you use certain cluster configurations, deploy your cluster in certain regions, or run multiple clusters from your account, you might need to request additional resources for your IBM Cloud account.

For a comprehensive list of the default IBM Cloud VPC quotas and service limits, see IBM Cloud's documentation for [Quotas and service limits](#).

Virtual Private Cloud (VPC)

Each OpenShift Container Platform cluster creates its own VPC. The default quota of VPCs per region is 10 and will allow 10 clusters. To have more than 10 clusters in a single region, you must increase this quota.

Application load balancer

By default, each cluster creates three application load balancers (ALBs):

- Internal load balancer for the master API server
- External load balancer for the master API server
- Load balancer for the router

You can create additional **LoadBalancer** service objects to create additional ALBs. The default quota of VPC ALBs are 50 per region. To have more than 50 ALBs, you must increase this quota.

VPC ALBs are supported. Classic ALBs are not supported for IBM Cloud VPC.

Floating IP address

By default, the installation program distributes control plane and compute machines across all availability zones within a region to provision the cluster in a highly available configuration. In each availability zone, a public gateway is created and requires a separate floating IP address.

The default quota for a floating IP address is 20 addresses per availability zone. The default cluster configuration yields three floating IP addresses:

- Two floating IP addresses in the **us-east-1** primary zone. The IP address associated with the bootstrap node is removed after installation.
- One floating IP address in the **us-east-2** secondary zone.
- One floating IP address in the **us-east-3** secondary zone.

IBM Cloud VPC can support up to 19 clusters per region in an account. If you plan to have more than 19 default clusters, you must increase this quota.

Virtual Server Instances (VSI)

By default, a cluster creates VSIs using **bx2-4x16** profiles, which includes the following resources by default:

- 4 vCPUs
- 16 GB RAM

The following nodes are created:

- One **bx2-4x16** bootstrap machine, which is removed after the installation is complete
- Three **bx2-4x16** control plane nodes
- Three **bx2-4x16** compute nodes

For more information, see IBM Cloud's documentation on [supported profiles](#).

Table 10.1. VSI component quotas and limits

VSI component	Default IBM Cloud VPC quota	Default cluster configuration	Maximum number of clusters
vCPU	200 vCPUs per region	28 vCPUs, or 24 vCPUs after bootstrap removal	8 per region
RAM	1600 GB per region	112 GB, or 96 GB after bootstrap removal	16 per region
Storage	18 TB per region	1050 GB, or 900 GB after bootstrap removal	19 per region

If you plan to exceed the resources stated in the table, you must increase your IBM Cloud account quota.

Block Storage Volumes

For each VPC machine, a block storage device is attached for its boot volume. The default cluster configuration creates seven VPC machines, resulting in seven block storage volumes. Additional Kubernetes persistent volume claims (PVCs) of the IBM Cloud VPC storage class create additional block storage volumes. The default quota of VPC block storage volumes are 300 per region. To have more than 300 volumes, you must increase this quota.

10.2.3. Configuring DNS resolution

How you configure DNS resolution depends on the type of OpenShift Container Platform cluster you are installing:

- If you are installing a public cluster, you use IBM Cloud Internet Services (CIS).
- If you are installing a private cluster, you use IBM Cloud DNS Services (DNS Services)

10.2.3.1. Using IBM Cloud Internet Services for DNS resolution

The installation program uses IBM Cloud Internet Services (CIS) to configure cluster DNS resolution and provide name lookup for a public cluster.



NOTE

IBM Cloud VPC does not support IPv6, so dual stack or IPv6 environments are not possible.

You must create a domain zone in CIS in the same account as your cluster. You must also ensure the zone is authoritative for the domain. You can do this using a root domain or subdomain.

Prerequisites

- You have installed the [IBM Cloud CLI](#).
- You have an existing domain and registrar. For more information, see the IBM [documentation](#).

Procedure

1. Create a CIS instance to use with your cluster:

a. Install the CIS plugin:

```
$ ibmcloud plugin install cis
```

b. Create the CIS instance:

```
$ ibmcloud cis instance-create <instance_name> standard 1
```

1 At a minimum, a **Standard** plan is required for CIS to manage the cluster subdomain and its DNS records.

2. Connect an existing domain to your CIS instance:

a. Set the context instance for CIS:

```
$ ibmcloud cis instance-set <instance_name> 1
```

1 The instance cloud resource name.

b. Add the domain for CIS:

```
$ ibmcloud cis domain-add <domain_name> 1
```

- 1 The fully qualified domain name. You can use either the root domain or subdomain value as the domain name, depending on which you plan to configure.



NOTE

A root domain uses the form **openshiftcorp.com**. A subdomain uses the form **clusters.openshiftcorp.com**.

3. Open the [CIS web console](#), navigate to the **Overview** page, and note your CIS name servers. These name servers will be used in the next step.
4. Configure the name servers for your domains or subdomains at the domain's registrar or DNS provider. For more information, see IBM Cloud's [documentation](#).

10.2.3.2. Using IBM Cloud DNS Services for DNS resolution

The installation program uses IBM Cloud DNS Services to configure cluster DNS resolution and provide name lookup for a private cluster.

You configure DNS resolution by creating a DNS services instance for the cluster, and then adding a DNS zone to the DNS Services instance. Ensure that the zone is authoritative for the domain. You can do this using a root domain or subdomain.



NOTE

IBM Cloud VPC does not support IPv6, so dual stack or IPv6 environments are not possible.

Prerequisites

- You have installed the [IBM Cloud CLI](#).
- You have an existing domain and registrar. For more information, see the [IBM documentation](#).

Procedure

1. Create a DNS Services instance to use with your cluster:
 - a. Install the DNS Services plugin by running the following command:

```
$ ibmcloud plugin install cloud-dns-services
```

- b. Create the DNS Services instance by running the following command:

```
$ ibmcloud dns instance-create <instance-name> standard-dns 1
```

- 1 At a minimum, a **Standard** plan is required for DNS Services to manage the cluster subdomain and its DNS records.

2. Create a DNS zone for the DNS Services instance:
 - a. Set the target operating DNS Services instance by running the following command:

```
$ ibmcloud dns instance-target <instance-name>
```

- b. Add the DNS zone to the DNS Services instance by running the following command:

```
$ ibmcloud dns zone-create <zone-name> 1
```

- 1** The fully qualified zone name. You can use either the root domain or subdomain value as the zone name, depending on which you plan to configure. A root domain uses the form **openshiftcorp.com**. A subdomain uses the form **clusters.openshiftcorp.com**.

3. Record the name of the DNS zone you have created. As part of the installation process, you must update the **install-config.yaml** file before deploying the cluster. Use the name of the DNS zone as the value for the **baseDomain** parameter.



NOTE

You do not have to manage permitted networks or configure an "A" DNS resource record. As required, the installation program configures these resources automatically.

10.2.4. IBM Cloud VPC IAM Policies and API Key

To install OpenShift Container Platform into your IBM Cloud account, the installation program requires an IAM API key, which provides authentication and authorization to access IBM Cloud service APIs. You can use an existing IAM API key that contains the required policies or create a new one.

For an IBM Cloud IAM overview, see the IBM Cloud [documentation](#).

10.2.4.1. Required access policies

You must assign the required access policies to your IBM Cloud account.

Table 10.2. Required access policies

Service type	Service	Access policy scope	Platform access	Service access
Account management	IAM Identity Service	All resources or a subset of resources ^[1]	Editor, Operator, Viewer, Administrator	Service ID creator
Account management ^[2]	Identity and Access Management	All resources	Editor, Operator, Viewer, Administrator	
Account management	Resource group only	All resource groups in the account	Administrator	

Service type	Service	Access policy scope	Platform access	Service access
IAM services	Cloud Object Storage	All resources or a subset of resources ^[1]	Editor, Operator, Viewer, Administrator	Reader, Writer, Manager, Content Reader, Object Reader, Object Writer
IAM services	Internet Services	All resources or a subset of resources ^[1]	Editor, Operator, Viewer, Administrator	Reader, Writer, Manager
IAM services	DNS Services	All resources or a subset of resources ^[1]	Editor, Operator, Viewer, Administrator	Reader, Writer, Manager
IAM services	VPC Infrastructure Services	All resources or a subset of resources ^[1]	Editor, Operator, Viewer, Administrator	Reader, Writer, Manager

1. The policy access scope should be set based on how granular you want to assign access. The scope can be set to **All resources** or **Resources based on selected attributes**
2. Optional: This access policy is only required if you want the installation program to create a resource group. For more information about resource groups, see the IBM [documentation](#).

10.2.4.2. Access policy assignment

In IBM Cloud VPC IAM, access policies can be attached to different subjects:

- Access group (Recommended)
- Service ID
- User

The recommended method is to define IAM access policies in an [access group](#). This helps organize all the access required for OpenShift Container Platform and enables you to onboard users and service IDs to this group. You can also assign access to [users and service IDs](#) directly, if desired.

10.2.4.3. Creating an API key

You must create a user API key or a service ID API key for your IBM Cloud account.

Prerequisites

- You have assigned the required access policies to your IBM Cloud account.
- You have attached your IAM access policies to an access group, or other appropriate resource.

Procedure

- Create an API key, depending on how you defined your IAM access policies. For example, if you assigned your access policies to a user, you must create a [user API key](#). If you assigned your access policies to a service ID, you must create a [service ID API key](#). If your access policies are assigned to an access group, you can use either API key type. For more information on IBM Cloud VPC API keys, see [Understanding API keys](#).

10.2.5. Supported IBM Cloud VPC regions

You can deploy an OpenShift Container Platform cluster to the following regions:

- **au-syd** (Sydney, Australia)
- **br-sao** (Sao Paulo, Brazil)
- **ca-tor** (Toronto, Canada)
- **eu-de** (Frankfurt, Germany)
- **eu-gb** (London, United Kingdom)
- **jp-osa** (Osaka, Japan)
- **jp-tok** (Tokyo, Japan)
- **us-east** (Washington DC, United States)
- **us-south** (Dallas, United States)

10.2.6. Next steps

- [Configuring IAM for IBM Cloud VPC](#)

10.3. CONFIGURING IAM FOR IBM CLOUD VPC

In environments where the cloud identity and access management (IAM) APIs are not reachable, you must put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

10.3.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the **credentialsMode** parameter in the **install-config.yaml** file.

Storing an administrator-level credential secret in the cluster **kube-system** project is not supported for IBM Cloud; therefore, you must set the **credentialsMode** parameter for the CCO to **Manual** when installing OpenShift Container Platform and manage your cloud credentials manually.

Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

Additional resources

- [About the Cloud Credential Operator](#)

10.3.2. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.



NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



NOTE

Ensure that the architecture of the **\$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE --file="/usr/bin/ccoctl" -a ~/.pull-secret
```

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl
```

Verification

- To verify that **ccoctl** is ready to use, display the help file by running the following command:

```
$ ccoctl --help
```

Output of **ccoctl --help**

OpenShift credentials provisioning tool

Usage:

```
ccoctl [command]
```

Available Commands:

```
alibabacloud  Manage credentials objects for alibaba cloud
aws           Manage credentials objects for AWS cloud
gcp           Manage credentials objects for Google cloud
help          Help about any command
ibmcloud      Manage credentials objects for IBM Cloud
nutanix       Manage credentials objects for Nutanix
```

Flags:

```
-h, --help  help for ccoctl
```

Use "ccoctl [command] --help" for more information about a command.

Additional resources

- [Rotating API keys for IBM Cloud VPC](#)

10.3.3. Next steps

- [Installing a cluster on IBM Cloud VPC with customizations](#)

10.3.4. Additional resources

- [Preparing to update a cluster with manually maintained credentials](#)

10.4. INSTALLING A CLUSTER ON IBM CLOUD VPC WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on infrastructure that the installation program provisions on IBM Cloud VPC. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

10.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an IBM Cloud account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You configured the **ccoctl** utility before you installed the cluster. For more information, see [Configuring IAM for IBM Cloud VPC](#).

10.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

10.4.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

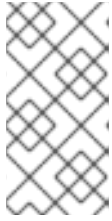
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.4.4. Obtaining the installation program

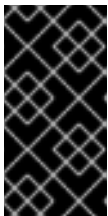
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

10.4.5. Exporting the IBM Cloud VPC API key

You must set the IBM Cloud VPC API key you created as a global variable; the installation program ingests the variable during startup to set the API key.

Prerequisites

- You have created either a user API key or service ID API key for your IBM Cloud account.

Procedure

- Export your IBM Cloud VPC API key as a global variable:

```
$ export IC_API_KEY=<api_key>
```



IMPORTANT

You must set the variable name exactly as specified; the installation program expects the variable name to be present during startup.

10.4.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on IBM Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

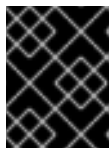
- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **ibmcloud** as the platform to target.
 - iii. Select the region to deploy the cluster to.
 - iv. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - v. Enter a descriptive name for your cluster.
 - vi. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

10.4.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

10.4.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 10.3. Required parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

10.4.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 10.4. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


10.4.6.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 10.5. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hypertreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; width: 65px; height: 360px; margin-bottom: 10px;"></div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="488 1440 592 1637" style="background-color: black; width: 65px; height: 88px; margin-bottom: 10px;"></div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

10.4.6.1.4. Additional IBM Cloud VPC configuration parameters

Additional IBM Cloud VPC configuration parameters are described in the following table:

Table 10.6. Additional IBM Cloud VPC parameters

Parameter	Description	Values
platform.ibmcloud.resourceGroupName	The name of an existing resource group to install your cluster to. This resource group must only be used for this specific cluster because the cluster components assume ownership of all of the resources in the resource group. If undefined, a new resource group is created for the cluster. [1]	String, for example existing_resource_group .
platform.ibmcloud.dedicatedHosts.profile	The new dedicated host to create. If you specify a value for platform.ibmcloud.dedicatedHosts.name , this parameter is not required.	Valid IBM Cloud VPC dedicated host profile, such as cx2-host-152x304 . [2]

Parameter	Description	Values
platform.ibmcloud.dedicatedHosts.name	An existing dedicated host. If you specify a value for platform.ibmcloud.dedicatedHosts.profile , this parameter is not required.	String, for example my-dedicated-host-name .
platform.ibmcloud.type	The instance type for all IBM Cloud VPC machines.	Valid IBM Cloud VPC instance type, such as bx2-8x32 . [2]
platform.ibmcloud.vpcName	The name of the existing VPC that you want to deploy your cluster to.	String.
platform.ibmcloud.controlPlaneSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your control plane machines to. Specify a subnet for each availability zone.	String array
platform.ibmcloud.computeSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your compute machines to. Specify a subnet for each availability zone. Subnet IDs are not supported.	String array

1. Whether you define an existing resource group, or if the installer creates one, determines how the resource group is treated when the cluster is uninstalled. If you define a resource group, the installer removes all of the installer-provisioned resources, but leaves the resource group alone; if a resource group is created as part of the installation, the installer removes all of the installer-provisioned resources and the resource group.
2. To determine which profile best meets your needs, see [Instance Profiles](#) in the IBM documentation.

10.4.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 10.7. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

10.4.6.3. Sample customized install-config.yaml file for IBM Cloud VPC

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and then modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: 5 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
  - 172.30.0.0/16

```

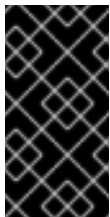
```
platform:
  ibmcloud:
    region: us-south 10
  credentialsMode: Manual
  publish: External
  pullSecret: '{"auths": ...}' 11
  fips: false 12
  sshKey: ssh-ed25519 AAAA... 13
```

1 8 10 11 Required. The installation program prompts you for this value.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

3 6 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 7 Enables or disables simultaneous multithreading, also known as Hyper-Threading. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

12 Enables or disables FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

13 Optional: provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

10.4.6.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

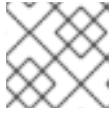
Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.

- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

10.4.7. Manually creating IAM for IBM Cloud VPC

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets for your cloud provider.

You can use the Cloud Credential Operator (CCO) utility (**ccoctl**) to create the required IBM Cloud VPC resources.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an existing **install-config.yaml** file.

Procedure

1. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example **install-config.yaml** configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

2. To generate the manifests, run the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

3. From the directory that contains the installation program, obtain the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. Extract the **CredentialsRequest** objects from the OpenShift Container Platform release image:

```
$ oc adm release extract --cloud=ibmcloud --credentials-requests $RELEASE_IMAGE \
--to=<path_to_credential_requests_directory> 1
```

- 1** The directory where the credential requests will be stored.

This command creates a YAML file for each **CredentialsRequest** object.

Sample **CredentialsRequest** object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
```



```

- attributes:
  - name: serviceName
    value: cloud-object-storage
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer
  - crn:v1:bluemix:public:iam::::role:Operator
  - crn:v1:bluemix:public:iam::::role:Editor
  - crn:v1:bluemix:public:iam::::serviceRole:Reader
  - crn:v1:bluemix:public:iam::::serviceRole:Writer
- attributes:
  - name: resourceType
    value: resource-group
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer

```

5. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on IBM Cloud VPC

```

0000_26_cloud-controller-manager-operator_15_credentialsrequest-ibm.yaml 1
0000_30_machine-api-operator_00_credentials-request.yaml 2
0000_50_cluster-image-registry-operator_01-registry-credentials-request-ibmcos.yaml 3
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 4
0000_50_cluster-storage-operator_03_credentials_request_ibm.yaml 5

```

- 1 The Cloud Controller Manager Operator CR is required.
- 2 The Machine API Operator CR is required.
- 3 The Image Registry Operator CR is required.
- 4 The Ingress Operator CR is required.
- 5 The Storage Operator CR is an optional component and might be disabled in your cluster.

6. Create the service ID for each credential request, assign the policies defined, create an API key in IBM Cloud VPC, and generate the secret:

```

$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir <path_to_credential_requests_directory> \ 1
  --name <cluster_name> \ 2
  --output-dir <installation_directory> \
  --resource-group-name <resource_group_name> 3

```

- 1 The directory where the credential requests are stored.
- 2 The name of the OpenShift Container Platform cluster.
- 3 Optional: The name of the resource group used for scoping the access policies.

**NOTE**

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

If an incorrect resource group name is provided, the installation fails during the bootstrap phase. To find the correct resource group name, run the following command:

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

Verification

- Ensure that the appropriate secrets were generated in your cluster's **manifests** directory.

10.4.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

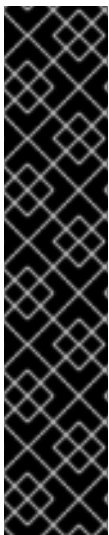
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

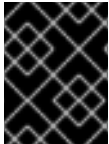
```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

10.4.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

10.4.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- [Accessing the web console](#)

10.4.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

10.4.12. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

10.5. INSTALLING A CLUSTER ON IBM CLOUD VPC WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on IBM Cloud VPC. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

10.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an IBM Cloud account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You configured the **ccoctl** utility before you installed the cluster. For more information, see [Configuring IAM for IBM Cloud VPC](#).

10.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

10.5.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```


**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.5.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

10.5.5. Exporting the IBM Cloud VPC API key

You must set the IBM Cloud VPC API key you created as a global variable; the installation program ingests the variable during startup to set the API key.

Prerequisites

- You have created either a user API key or service ID API key for your IBM Cloud account.

Procedure

- Export your IBM Cloud VPC API key as a global variable:

```
$ export IC_API_KEY=<api_key>
```



IMPORTANT

You must set the variable name exactly as specified; the installation program expects the variable name to be present during startup.

10.5.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on IBM Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.

- a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
- Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- Select **ibmcloud** as the platform to target.
 - Select the region to deploy the cluster to.
 - Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - Enter a descriptive name for your cluster.
 - Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
- Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 - Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

10.5.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's

platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

10.5.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 10.8. Required parameters

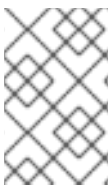
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

10.5.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 10.9. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


10.5.6.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 10.10. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1440 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

10.5.6.1.4. Additional IBM Cloud VPC configuration parameters

Additional IBM Cloud VPC configuration parameters are described in the following table:

Table 10.11. Additional IBM Cloud VPC parameters

Parameter	Description	Values
platform.ibmcloud.resourceGroupName	The name of an existing resource group to install your cluster to. This resource group must only be used for this specific cluster because the cluster components assume ownership of all of the resources in the resource group. If undefined, a new resource group is created for the cluster. [1]	String, for example existing_resource_group .
platform.ibmcloud.dedicatedHosts.profile	The new dedicated host to create. If you specify a value for platform.ibmcloud.dedicatedHosts.name , this parameter is not required.	Valid IBM Cloud VPC dedicated host profile, such as cx2-host-152x304 . [2]

Parameter	Description	Values
platform.ibmcloud.dedicatedHosts.name	An existing dedicated host. If you specify a value for platform.ibmcloud.dedicatedHosts.profile , this parameter is not required.	String, for example my-dedicated-host-name .
platform.ibmcloud.type	The instance type for all IBM Cloud VPC machines.	Valid IBM Cloud VPC instance type, such as bx2-8x32 . [2]
platform.ibmcloud.vpcName	The name of the existing VPC that you want to deploy your cluster to.	String.
platform.ibmcloud.controlPlaneSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your control plane machines to. Specify a subnet for each availability zone.	String array
platform.ibmcloud.computeSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your compute machines to. Specify a subnet for each availability zone. Subnet IDs are not supported.	String array

1. Whether you define an existing resource group, or if the installer creates one, determines how the resource group is treated when the cluster is uninstalled. If you define a resource group, the installer removes all of the installer-provisioned resources, but leaves the resource group alone; if a resource group is created as part of the installation, the installer removes all of the installer-provisioned resources and the resource group.
2. To determine which profile best meets your needs, see [Instance Profiles](#) in the IBM documentation.

10.5.6.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 10.12. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

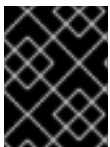
If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

10.5.6.3. Sample customized install-config.yaml file for IBM Cloud VPC

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and then modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: 5 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking: 9
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 10
  serviceNetwork:
  - 172.30.0.0/16

```

```

platform:
  ibmcloud:
    region: us-south 11
  credentialsMode: Manual
  publish: External
  pullSecret: '{"auths": ...}' 12
  fips: false 13
  sshKey: ssh-ed25519 AAAA... 14

```

1 8 11 12 Required. The installation program prompts you for this value.

2 5 9 If you do not provide these parameters and values, the installation program provides the default value.

3 6 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 7 Enables or disables simultaneous multithreading, also known as Hyper-Threading. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

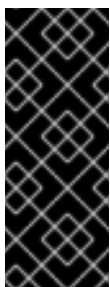


IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

10 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

13 Enables or disables FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

14 Optional: provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

10.5.6.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.

- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

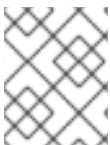
**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

10.5.7. Manually creating IAM for IBM Cloud VPC

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets for your cloud provider.

You can use the Cloud Credential Operator (CCO) utility (**ccoctl**) to create the required IBM Cloud VPC resources.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an existing **install-config.yaml** file.

Procedure

1. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example **install-config.yaml** configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

2. To generate the manifests, run the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

3. From the directory that contains the installation program, obtain the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. Extract the **CredentialsRequest** objects from the OpenShift Container Platform release image:

```
$ oc adm release extract --cloud=ibmcloud --credentials-requests $RELEASE_IMAGE \
--to=<path_to_credential_requests_directory> 1
```

- 1** The directory where the credential requests will be stored.

This command creates a YAML file for each **CredentialsRequest** object.

Sample **CredentialsRequest** object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
```

```

- attributes:
  - name: serviceName
    value: cloud-object-storage
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer
  - crn:v1:bluemix:public:iam::::role:Operator
  - crn:v1:bluemix:public:iam::::role:Editor
  - crn:v1:bluemix:public:iam::::serviceRole:Reader
  - crn:v1:bluemix:public:iam::::serviceRole:Writer
- attributes:
  - name: resourceType
    value: resource-group
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer

```

5. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on IBM Cloud VPC

```

0000_26_cloud-controller-manager-operator_15_credentialsrequest-ibm.yaml 1
0000_30_machine-api-operator_00_credentials-request.yaml 2
0000_50_cluster-image-registry-operator_01-registry-credentials-request-ibmcos.yaml 3
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 4
0000_50_cluster-storage-operator_03_credentials_request_ibm.yaml 5

```

- 1 The Cloud Controller Manager Operator CR is required.
- 2 The Machine API Operator CR is required.
- 3 The Image Registry Operator CR is required.
- 4 The Ingress Operator CR is required.
- 5 The Storage Operator CR is an optional component and might be disabled in your cluster.

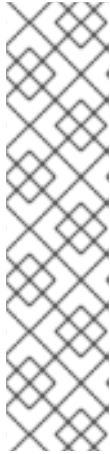
6. Create the service ID for each credential request, assign the policies defined, create an API key in IBM Cloud VPC, and generate the secret:

```

$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir <path_to_credential_requests_directory> \ 1
  --name <cluster_name> \ 2
  --output-dir <installation_directory> \
  --resource-group-name <resource_group_name> 3

```

- 1 The directory where the credential requests are stored.
- 2 The name of the OpenShift Container Platform cluster.
- 3 Optional: The name of the resource group used for scoping the access policies.

**NOTE**

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

If an incorrect resource group name is provided, the installation fails during the bootstrap phase. To find the correct resource group name, run the following command:

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

Verification

- Ensure that the appropriate secrets were generated in your cluster's **manifests** directory.

10.5.8. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.

**NOTE**

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

**IMPORTANT**

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

10.5.9. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

- Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

- Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following example:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

- Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

10.5.10. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

10.5.10.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 10.13. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 10.14. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 10.15. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 10.16. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

**NOTE**

IPsec for the OVN-Kubernetes network plugin is not supported when installing a cluster on IBM Cloud.

Table 10.17. policyAuditConfig object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 10.18. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 10.19. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

10.5.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

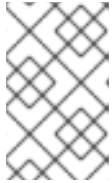
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

10.5.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:


```
$ oc <command>
```

10.5.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- [Accessing the web console](#)

10.5.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

10.5.15. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

10.6. INSTALLING A CLUSTER ON IBM CLOUD VPC INTO AN EXISTING VPC

In OpenShift Container Platform version 4.12, you can install a cluster into an existing Virtual Private Cloud (VPC) on IBM Cloud VPC. The installation program provisions the rest of the required infrastructure, which you can then further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

10.6.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [configured an IBM Cloud account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You configured the **ccoctl** utility before you installed the cluster. For more information, see [Configuring IAM for IBM Cloud VPC](#).

10.6.2. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into the subnets of an existing IBM Virtual Private Cloud (VPC). Deploying OpenShift Container Platform into an existing VPC can help you avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are in your existing subnets, it cannot choose subnet CIDRs and so forth. You must configure networking for the subnets to which you will install the cluster.

10.6.2.1. Requirements for using your VPC

You must correctly configure the existing VPC and its subnets before you install the cluster. The installation program does not create the following components:

- NAT gateways
- Subnets
- Route tables
- VPC network

The installation program cannot:

- Subdivide network ranges for the cluster to use
- Set route tables for the subnets
- Set VPC options like DHCP

**NOTE**

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

10.6.2.2. VPC validation

The VPC and all of the subnets must be in an existing resource group. The cluster is deployed to this resource group.

As part of the installation, specify the following in the **install-config.yaml** file:

- The name of the resource group
- The name of VPC
- The subnets for control plane machines and compute machines

To ensure that the subnets that you provide are suitable, the installation program confirms the following:

- All of the subnets that you specify exist.
- For each availability zone in the region, you specify:
 - One subnet for control plane machines.
 - One subnet for compute machines.
- The machine CIDR that you specified contains the subnets for the compute machines and control plane machines.

**NOTE**

Subnet IDs are not supported.

10.6.2.3. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed to the entire network.
- TCP port 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

10.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

10.6.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.6.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

10.6.6. Exporting the IBM Cloud VPC API key

You must set the IBM Cloud VPC API key you created as a global variable; the installation program ingests the variable during startup to set the API key.

Prerequisites

- You have created either a user API key or service ID API key for your IBM Cloud account.

Procedure

- Export your IBM Cloud VPC API key as a global variable:

```
$ export IC_API_KEY=<api_key>
```



IMPORTANT

You must set the variable name exactly as specified; the installation program expects the variable name to be present during startup.

10.6.7. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on IBM Cloud.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them

into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **ibmcloud** as the platform to target.
 - iii. Select the region to deploy the cluster to.
 - iv. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.
 - v. Enter a descriptive name for your cluster.
 - vi. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

10.6.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

10.6.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 10.20. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

10.6.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 10.21. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <p>NOTE You cannot modify parameters specified by the networking object after installation.</p> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 . The CIDR must contain the subnets defined in platform.ibmcloud.controlPlaneSubnets and platform.ibmcloud.computeSubnets .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


10.6.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 10.22. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

10.6.7.1.4. Additional IBM Cloud VPC configuration parameters

Additional IBM Cloud VPC configuration parameters are described in the following table:

Table 10.23. Additional IBM Cloud VPC parameters

Parameter	Description	Values
platform.ibmcloud.resourceGroupName	The name of an existing resource group. The existing VPC and subnets should be in this resource group. Cluster installation resources are created in this resource group.	String, for example existing_resource_group .
platform.ibmcloud.dedicatedHosts.profile	The new dedicated host to create. If you specify a value for platform.ibmcloud.dedicatedHosts.name , this parameter is not required.	Valid IBM Cloud VPC dedicated host profile, such as cx2-host-152x304 . [1]

Parameter	Description	Values
platform.ibmcloud.dedicatedHosts.name	An existing dedicated host. If you specify a value for platform.ibmcloud.dedicatedHosts.profile , this parameter is not required.	String, for example my-dedicated-host-name .
platform.ibmcloud.type	The instance type for all IBM Cloud VPC machines.	Valid IBM Cloud VPC instance type, such as bx2-8x32 . [1]
platform.ibmcloud.vpcName	The name of the existing VPC that you want to deploy your cluster to.	String.
platform.ibmcloud.controlPlaneSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your control plane machines to. Specify a subnet for each availability zone.	String array
platform.ibmcloud.computeSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your compute machines to. Specify a subnet for each availability zone. Subnet IDs are not supported.	String array

1. To determine which profile best meets your needs, see [Instance Profiles](#) in the IBM documentation.

10.6.7.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 10.24. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

10.6.7.3. Sample customized install-config.yaml file for IBM Cloud VPC

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and then modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: 5 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 10
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:

```

```

region: eu-gb 11
resourceGroupName: eu-gb-example-network-rg 12
vpcName: eu-gb-example-network-1 13
controlPlaneSubnets: 14
- eu-gb-example-network-1-cp-eu-gb-1
- eu-gb-example-network-1-cp-eu-gb-2
- eu-gb-example-network-1-cp-eu-gb-3
computeSubnets: 15
- eu-gb-example-network-1-compute-eu-gb-1
- eu-gb-example-network-1-compute-eu-gb-2
- eu-gb-example-network-1-compute-eu-gb-3
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 16
fips: false 17
sshKey: ssh-ed25519 AAAA... 18

```

1 8 11 16 Required. The installation program prompts you for this value.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

3 6 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 7 Enables or disables simultaneous multithreading, also known as Hyper-Threading. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

9 The machine CIDR must contain the subnets for the compute machines and control plane machines.

10 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

12 The name of an existing resource group. The existing VPC and subnets should be in this resource group. The cluster is deployed to this resource group.

13 Specify the name of an existing VPC.

14 Specify the name of the existing subnets to which to deploy the control plane machines. The subnets must belong to the VPC that you specified. Specify a subnet for each availability zone in the region.

- 15 Specify the name of the existing subnets to which to deploy the compute machines. The subnets must belong to the VPC that you specified. Specify a subnet for each availability zone in the region.
- 17 Enables or disables FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 18 Optional: provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

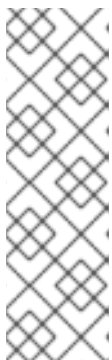
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

10.6.7.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
```

```

baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

10.6.8. Manually creating IAM for IBM Cloud VPC

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets for your cloud provider.

You can use the Cloud Credential Operator (CCO) utility (**ccoctl**) to create the required IBM Cloud VPC resources.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an existing **install-config.yaml** file.

Procedure

1. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

2. To generate the manifests, run the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

3. From the directory that contains the installation program, obtain the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk 'release image/ {print $3}')
```

4. Extract the **CredentialsRequest** objects from the OpenShift Container Platform release image:

```
$ oc adm release extract --cloud=ibmcloud --credentials-requests $RELEASE_IMAGE \
--to=<path_to_credential_requests_directory> 1
```

- 1** The directory where the credential requests will be stored.

This command creates a YAML file for each **CredentialsRequest** object.

Sample **CredentialsRequest** object

```

apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
      - attributes:
          - name: serviceName
            value: cloud-object-storage
        roles:
          - crn:v1:bluemix:public:iam::::role:Viewer
          - crn:v1:bluemix:public:iam::::role:Operator
          - crn:v1:bluemix:public:iam::::role:Editor
          - crn:v1:bluemix:public:iam::::serviceRole:Reader
          - crn:v1:bluemix:public:iam::::serviceRole:Writer
      - attributes:
          - name: resourceType
            value: resource-group
        roles:
          - crn:v1:bluemix:public:iam::::role:Viewer

```

5. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example **credrequests** directory contents for OpenShift Container Platform 4.12 on IBM Cloud VPC

```

0000_26_cloud-controller-manager-operator_15_credentialsrequest-ibm.yaml 1
0000_30_machine-api-operator_00_credentials-request.yaml 2
0000_50_cluster-image-registry-operator_01-registry-credentials-request-ibmcos.yaml 3
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 4
0000_50_cluster-storage-operator_03_credentials_request_ibm.yaml 5

```

- 1** The Cloud Controller Manager Operator CR is required.
- 2** The Machine API Operator CR is required.
- 3** The Image Registry Operator CR is required.
- 4** The Ingress Operator CR is required.

- 5 The Storage Operator CR is an optional component and might be disabled in your cluster.
6. Create the service ID for each credential request, assign the policies defined, create an API key in IBM Cloud VPC, and generate the secret:

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir <path_to_credential_requests_directory> \ 1
  --name <cluster_name> \ 2
  --output-dir <installation_directory> \
  --resource-group-name <resource_group_name> 3
```

- 1 The directory where the credential requests are stored.
- 2 The name of the OpenShift Container Platform cluster.
- 3 Optional: The name of the resource group used for scoping the access policies.



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

If an incorrect resource group name is provided, the installation fails during the bootstrap phase. To find the correct resource group name, run the following command:

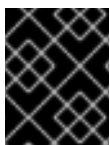
```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

Verification

- Ensure that the appropriate secrets were generated in your cluster's **manifests** directory.

10.6.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



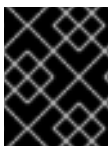
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

10.6.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

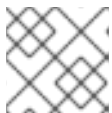
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

10.6.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- [Accessing the web console](#)

10.6.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

10.6.13. Next steps

- [Customize your cluster.](#)
- Optional: [Opt out of remote health reporting.](#)

10.7. INSTALLING A PRIVATE CLUSTER ON IBM CLOUD VPC

In OpenShift Container Platform version 4.12, you can install a private cluster into an existing VPC. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

10.7.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users.](#)
- You [configured an IBM Cloud account](#) to host the cluster.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.
- You configured the `ccoctl` utility before you installed the cluster. For more information, see [Configuring IAM for IBM Cloud VPC.](#)

10.7.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.



IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Create a DNS zone using IBM Cloud DNS Services and specify it as the base domain of the cluster. For more information, see "Using IBM Cloud DNS Services to configure DNS resolution".
- Deploy from a machine that has access to:

- The API services for the cloud to which you provision.
- The hosts on the network that you provision.
- The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

10.7.3. Private clusters in IBM Cloud VPC

To create a private cluster on IBM Cloud VPC, you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

The cluster still requires access to internet to access the IBM Cloud VPC APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public network load balancers, which support public ingress
- A public DNS zone that matches the **baseDomain** for the cluster

The installation program does use the **baseDomain** that you specify to create a private DNS zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

10.7.3.1. Limitations

Private clusters on IBM Cloud VPC are subject only to the limitations associated with the existing VPC that was used for cluster deployment.

10.7.4. About using a custom VPC

In OpenShift Container Platform 4.12, you can deploy a cluster into the subnets of an existing IBM Virtual Private Cloud (VPC). Deploying OpenShift Container Platform into an existing VPC can help you avoid limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are in your existing subnets, it cannot choose subnet CIDRs and so forth. You must configure networking for the subnets to which you will install the cluster.

10.7.4.1. Requirements for using your VPC

You must correctly configure the existing VPC and its subnets before you install the cluster. The installation program does not create the following components:

- NAT gateways
- Subnets

- Route tables
- VPC network

The installation program cannot:

- Subdivide network ranges for the cluster to use
- Set route tables for the subnets
- Set VPC options like DHCP



NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

10.7.4.2. VPC validation

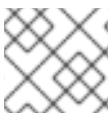
The VPC and all of the subnets must be in an existing resource group. The cluster is deployed to this resource group.

As part of the installation, specify the following in the **install-config.yaml** file:

- The name of the resource group
- The name of VPC
- The subnets for control plane machines and compute machines

To ensure that the subnets that you provide are suitable, the installation program confirms the following:

- All of the subnets that you specify exist.
- For each availability zone in the region, you specify:
 - One subnet for control plane machines.
 - One subnet for compute machines.
- The machine CIDR that you specified contains the subnets for the compute machines and control plane machines.



NOTE

Subnet IDs are not supported.

10.7.4.3. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed to the entire network.

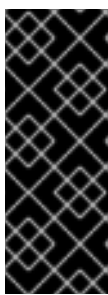
- TCP port 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

10.7.5. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

10.7.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

■

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.7.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a bastion host on your cloud network or a machine that has access to the to the network through a VPN.

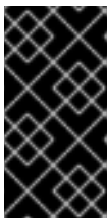
For more information about private cluster installation requirements, see "Private clusters".

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

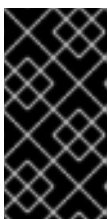
Procedure

- Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
- Select your infrastructure provider.
- Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

- Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

■

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

10.7.8. Exporting the IBM Cloud VPC API key

You must set the IBM Cloud VPC API key you created as a global variable; the installation program ingests the variable during startup to set the API key.

Prerequisites

- You have created either a user API key or service ID API key for your IBM Cloud account.

Procedure

- Export your IBM Cloud VPC API key as a global variable:

```
$ export IC_API_KEY=<api_key>
```



IMPORTANT

You must set the variable name exactly as specified; the installation program expects the variable name to be present during startup.

10.7.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

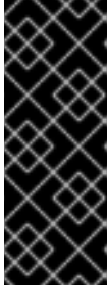
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

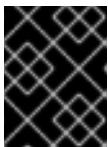
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

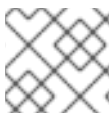
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

10.7.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

10.7.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 10.25. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

10.7.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 10.26. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 . The CIDR must contain the subnets defined in platform.ibmcloud.controlPlaneSubnets and platform.ibmcloud.computeSubnets .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


10.7.9.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 10.27. Optional parameters


Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1285" style="border: 1px solid #ccc; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1442 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	Internal or External . To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal . The default value is External .
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p>  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p>	For example, sshKey: ssh-ed25519 AAAA...

10.7.9.1.4. Additional IBM Cloud VPC configuration parameters

Additional IBM Cloud VPC configuration parameters are described in the following table:

Table 10.28. Additional IBM Cloud VPC parameters

Parameter	Description	Values
platform.ibmcloud.resourceGroupName	The name of an existing resource group. The existing VPC and subnets should be in this resource group. Cluster installation resources are created in this resource group.	String, for example existing_resource_group .
platform.ibmcloud.dedicatedHosts.profile	The new dedicated host to create. If you specify a value for platform.ibmcloud.dedicatedHosts.name , this parameter is not required.	Valid IBM Cloud VPC dedicated host profile, such as cx2-host-152x304 . [1]

Parameter	Description	Values
platform.ibmcloud.dedicatedHosts.name	An existing dedicated host. If you specify a value for platform.ibmcloud.dedicatedHosts.profile , this parameter is not required.	String, for example my-dedicated-host-name .
platform.ibmcloud.type	The instance type for all IBM Cloud VPC machines.	Valid IBM Cloud VPC instance type, such as bx2-8x32 . [1]
platform.ibmcloud.vpcName	The name of the existing VPC that you want to deploy your cluster to.	String.
platform.ibmcloud.controlPlaneSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your control plane machines to. Specify a subnet for each availability zone.	String array
platform.ibmcloud.computeSubnets	The name(s) of the existing subnet(s) in your VPC that you want to deploy your compute machines to. Specify a subnet for each availability zone. Subnet IDs are not supported.	String array

1. To determine which profile best meets your needs, see [Instance Profiles](#) in the IBM documentation.

10.7.9.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 10.29. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU	Virtual RAM	Storage	IOPS
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

10.7.9.3. Sample customized install-config.yaml file for IBM Cloud VPC

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and then modify it.

```

apiVersion: v1
baseDomain: example.com 1
controlPlane: 2 3
  hyperthreading: Enabled 4
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: 5 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 10
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:

```



```

ibmcloud:
  region: eu-gb 12
  resourceGroupName: eu-gb-example-network-rg 13
  vpcName: eu-gb-example-network-1 14
  controlPlaneSubnets: 15
    - eu-gb-example-network-1-cp-eu-gb-1
    - eu-gb-example-network-1-cp-eu-gb-2
    - eu-gb-example-network-1-cp-eu-gb-3
  computeSubnets: 16
    - eu-gb-example-network-1-compute-eu-gb-1
    - eu-gb-example-network-1-compute-eu-gb-2
    - eu-gb-example-network-1-compute-eu-gb-3
  credentialsMode: Manual
  publish: Internal 17
  pullSecret: '{"auths": ...}' 18
  fips: false 19
  sshKey: ssh-ed25519 AAAA... 20

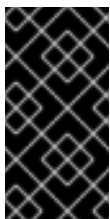
```

1 8 12 18 Required.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

3 6 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 7 Enables or disables simultaneous multithreading, also known as Hyper-Threading. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

9 The machine CIDR must contain the subnets for the compute machines and control plane machines.

10 The CIDR must contain the subnets defined in **platform.ibmcloud.controlPlaneSubnets** and **platform.ibmcloud.computeSubnets**.

11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

13 The name of an existing resource group. The existing VPC and subnets should be in this resource group. The cluster is deployed to this resource group.

14 Specify the name of an existing VPC.

- 15 Specify the name of the existing subnets to which to deploy the control plane machines. The subnets must belong to the VPC that you specified. Specify a subnet for each availability zone in
- 16 Specify the name of the existing subnets to which to deploy the compute machines. The subnets must belong to the VPC that you specified. Specify a subnet for each availability zone in the region.
- 17 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster. The default value is **External**.
- 19 Enables or disables FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 20 Optional: provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

10.7.9.4. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

10.7.10. Manually creating IAM for IBM Cloud VPC

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets for your cloud provider.

You can use the Cloud Credential Operator (CCO) utility (**ccoctl**) to create the required IBM Cloud VPC resources.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an existing **install-config.yaml** file.

Procedure

1. Edit the **install-config.yaml** configuration file so that it contains the **credentialsMode** parameter set to **Manual**.

Example **install-config.yaml** configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: amd64
  hyperthreading: Enabled
```

- 1** This line is added to set the **credentialsMode** parameter to **Manual**.

2. To generate the manifests, run the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

3. From the directory that contains the installation program, obtain the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk 'release image/' {print $3})
```

4. Extract the **CredentialsRequest** objects from the OpenShift Container Platform release image:

```
$ oc adm release extract --cloud=ibmcloud --credentials-requests $RELEASE_IMAGE \
--to=<path_to_credential_requests_directory> 1
```

- 1** The directory where the credential requests will be stored.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
      - attributes:
          - name: serviceName
            value: cloud-object-storage
        roles:
          - crn:v1:bluemix:public:iam::::role:Viewer
          - crn:v1:bluemix:public:iam::::role:Operator
          - crn:v1:bluemix:public:iam::::role:Editor
          - crn:v1:bluemix:public:iam::::serviceRole:Reader
          - crn:v1:bluemix:public:iam::::serviceRole:Writer
      - attributes:
          - name: resourceType
            value: resource-group
        roles:
          - crn:v1:bluemix:public:iam::::role:Viewer
```

5. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on IBM Cloud VPC

```
0000_26_cloud-controller-manager-operator_15_credentialsrequest-ibm.yaml 1
0000_30_machine-api-operator_00_credentials-request.yaml 2
0000_50_cluster-image-registry-operator_01-registry-credentials-request-ibmcos.yaml 3
0000_50_cluster-ingress-operator_00-ingress-credentials-request.yaml 4
0000_50_cluster-storage-operator_03_credentials_request_ibm.yaml 5
```

- 1** The Cloud Controller Manager Operator CR is required.

- 2 The Machine API Operator CR is required.
 - 3 The Image Registry Operator CR is required.
 - 4 The Ingress Operator CR is required.
 - 5 The Storage Operator CR is an optional component and might be disabled in your cluster.
6. Create the service ID for each credential request, assign the policies defined, create an API key in IBM Cloud VPC, and generate the secret:

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir <path_to_credential_requests_directory> \ 1
  --name <cluster_name> \ 2
  --output-dir <installation_directory> \
  --resource-group-name <resource_group_name> 3
```

- 1 The directory where the credential requests are stored.
- 2 The name of the OpenShift Container Platform cluster.
- 3 Optional: The name of the resource group used for scoping the access policies.



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

If an incorrect resource group name is provided, the installation fails during the bootstrap phase. To find the correct resource group name, run the following command:

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

Verification

- Ensure that the appropriate secrets were generated in your cluster's **manifests** directory.

10.7.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

10.7.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```


-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

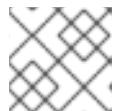
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

10.7.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- [Accessing the web console](#)

10.7.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- [About remote health monitoring](#)

10.7.15. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

10.8. UNINSTALLING A CLUSTER ON IBM CLOUD VPC

You can remove a cluster that you deployed to IBM Cloud VPC.

10.8.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.
- You have configured the **ccoctl** binary.
- You have installed the IBM Cloud CLI and installed or updated the VPC infrastructure service plugin. For more information see "Prerequisites" in the [IBM Cloud VPC CLI documentation](#) .

Procedure

1. If the following conditions are met, this step is required:
 - The installation program created a resource group as part of the installation process.
 - You or one of your applications created persistent volume claims (PVCs) after the cluster was deployed.

In which case, the PVCs are not removed when uninstalling the cluster, which might prevent the resource group from being successfully removed. To prevent a failure:

- a. Log in to the IBM Cloud using the CLI.
- b. To list the PVCs, run the following command:

```
$ ibmcloud is volumes --resource-group-name <infrastructure_id>
```

For more information about listing volumes, see the [IBM Cloud VPC CLI documentation](#) .

- c. To delete the PVCs, run the following command:

```
$ ibmcloud is volume-delete --force <volume_id>
```

For more information about deleting volumes, see the [IBM Cloud VPC CLI documentation](#).

- Export the IBM Cloud API key that was created as part of the installation process.

```
$ export IC_API_KEY=<api_key>
```



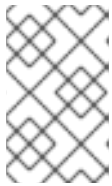
NOTE

You must set the variable name exactly as specified. The installation program expects the variable name to be present to remove the service IDs that were created when the cluster was installed.

- On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$ ./openshift-install destroy cluster \  
--dir <installation_directory> --log-level info 1 2
```

- For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

- Remove the manual CCO credentials that were created for the cluster:

```
$ ccoctl ibmcloud delete-service-id \  
--credentials-requests-dir <path_to_credential_requests_directory> \  
--name <cluster_name>
```



NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

- Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 11. INSTALLING ON NUTANIX

11.1. PREPARING TO INSTALL ON NUTANIX

Before you install an OpenShift Container Platform cluster, be sure that your Nutanix environment meets the following requirements.

11.1.1. Nutanix version requirements

You must install the OpenShift Container Platform cluster to a Nutanix environment that meets the following requirements.

Table 11.1. Version requirements for Nutanix virtual environments

Component	Required version
Nutanix AOS	5.20.4+ or 6.5.1+
Prism Central	2022.4+

11.1.2. Environment requirements

Before you install an OpenShift Container Platform cluster, review the following Nutanix AOS environment requirements.

11.1.2.1. Required account privileges

The installation program requires access to a Nutanix account with the necessary permissions to deploy the cluster and to maintain the daily operation of it. The following options are available to you:

- You can use a local Prism Central user account with administrative privileges. Using a local account is the quickest way to grant access to an account with the required permissions.
- If your organization's security policies require that you use a more restrictive set of permissions, use the permissions that are listed in the following table to create a custom Cloud Native role in Prism Central. You can then assign the role to a user account that is a member of a Prism Central authentication directory. When assigning entities to the role, ensure that the user can access only the Prism Element and subnet that are required to deploy the virtual machines. For more information, see the Nutanix documentation about creating a [Custom Cloud Native role](#) and [assigning a role](#).

Example 11.1. Required permissions for creating a Custom Cloud Native role

Nutanix Object	Required permissions in Nutanix API	Description
----------------	-------------------------------------	-------------

Nutanix Object	Required permissions in Nutanix API	Description
Categories	Create_Category_Mapping Create_Or_Update_Name_Category Create_Or_Update_Value_Category Delete_Category_Mapping Delete_Name_Category Delete_Value_Category View_Category_Mapping View_Name_Category View_Value_Category	Create, read, and delete categories that are assigned to the OpenShift Container Platform machines.
Images	Create_Image Delete_Image View_Image	Create, read, and delete the operating system images used for the OpenShift Container Platform machines.
Virtual Machines	Create_Virtual_Machine Delete_Virtual_Machine View_Virtual_Machine	Create, read, and delete the OpenShift Container Platform machines.
Clusters	View_Cluster	View the Prism Element clusters that host the OpenShift Container Platform machines.
Subnets	View_Subnet	View the subnets that host the OpenShift Container Platform machines.

11.1.2.2. Cluster limits

Available resources vary between clusters. The number of possible clusters within a Nutanix environment is limited primarily by available storage space and any limitations associated with the resources that the cluster creates, and resources that you require to deploy the cluster, such as IP addresses and networks.

11.1.2.3. Cluster resources

A minimum of 800 GB of storage is required to use a standard cluster.

When you deploy a OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your Nutanix instance. Although these resources use 856 GB of storage, the bootstrap node is destroyed as part of the installation process.

A standard OpenShift Container Platform installation creates the following resources:

- 1 label

- Virtual machines:
 - 1 disk image
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

11.1.2.4. Networking requirements

You must use AHV IP Address Management (IPAM) for the network and ensure that it is configured to provide persistent IP addresses to the cluster machines. Additionally, create the following networking resources before you install the OpenShift Container Platform cluster:

- IP addresses
- DNS records



NOTE

It is recommended that each OpenShift Container Platform node in the cluster have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, an NTP server prevents errors typically associated with asynchronous server clocks.

11.1.2.4.1. Required IP Addresses

An installer-provisioned installation requires two static virtual IP (VIP) addresses:

- A VIP address for the API is required. This address is used to access the cluster API.
- A VIP address for ingress is required. This address is used for cluster ingress traffic.

You specify these IP addresses when you install the OpenShift Container Platform cluster.

11.1.2.4.2. DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the Nutanix instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster.

If you use your own DNS or DHCP server, you must also create records for each node, including the bootstrap, control plane, and compute nodes.

A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 11.2. Required DNS records

Component	Record	Description

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

11.1.3. Configuring the Cloud Credential Operator utility

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). To install a cluster on Nutanix, you must set the CCO to **manual** mode as part of the installation process.

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccocctl**) binary.



NOTE

The **ccocctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```


**NOTE**

Ensure that the architecture of the **\$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE --file="/usr/bin/ccoctl" -a ~/.pull-secret
```

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl
```

Verification

- To verify that **ccoctl** is ready to use, display the help file by running the following command:

```
$ ccoctl --help
```

Output of ccoctl --help

```
OpenShift credentials provisioning tool
```

```
Usage:
```

```
ccoctl [command]
```

```
Available Commands:
```

```
alibabacloud Manage credentials objects for alibaba cloud
```

```
aws          Manage credentials objects for AWS cloud
```

```
gcp         Manage credentials objects for Google cloud
```

```
help       Help about any command
```

```
ibmcloud   Manage credentials objects for IBM Cloud
```

```
nutanix    Manage credentials objects for Nutanix
```

```
Flags:
```

```
-h, --help  help for ccoctl
```

```
Use "ccoctl [command] --help" for more information about a command.
```

Additional resources

- [Preparing to update a cluster with manually maintained credentials](#)

11.2. INSTALLING A CLUSTER ON NUTANIX

In OpenShift Container Platform version 4.12, you can choose one of the following options to install a cluster on your Nutanix instance:

Using installer-provisioned infrastructure Use the procedures in the following sections to use installer-provisioned infrastructure. Installer-provisioned infrastructure is ideal for installing in connected or disconnected network environments. The installer-provisioned infrastructure includes an installation program that provisions the underlying infrastructure for the cluster.

Using the Assisted Installer. The [Assisted Installer](#) hosted at console.redhat.com. The Assisted Installer cannot be used in disconnected environments. The Assisted Installer does not provision the underlying infrastructure for the cluster, so you must provision the infrastructure before the running the Assisted Installer. Installing with the Assisted Installer also provides integration with Nutanix, enabling autoscaling. See [Installing an on-premise cluster using the Assisted Installer](#) for additional details.

Using user-provisioned infrastructure Complete the relevant steps outlined in the [Installing a cluster on any platform](#) documentation.

11.2.1. Prerequisites

- You have reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- The installation program requires access to port 9440 on Prism Central and Prism Element. You verified that port 9440 is accessible.
- If you use a firewall, you have met these prerequisites:
 - You confirmed that port 9440 is accessible. Control plane nodes must be able to reach Prism Central and Prism Element on port 9440 for the installation to succeed.
 - You configured the firewall to [grant access](#) to the sites that OpenShift Container Platform requires. This includes the use of Telemetry.
- If your Nutanix environment is using the default self-signed SSL certificate, replace it with a certificate that is signed by a CA. The installation program requires a valid CA-signed certificate to access to the Prism Central API. For more information about replacing the self-signed certificate, see the [Nutanix AOS Security Guide](#).

If your Nutanix environment uses an internal CA to issue certificates, you must configure a cluster-wide proxy as part of the installation process. For more information, see [Configuring a custom PKI](#).



IMPORTANT

Use 2048-bit certificates. The installation fails if you use 4096-bit certificates with Prism Central 2022.x.

11.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

11.2.3. Internet access for Prism Central

Prism Central requires internet access to obtain the Red Hat Enterprise Linux CoreOS (RHCOS) image that is required to install the cluster. The RHCOS image for Nutanix is available at rhcos.mirror.openshift.com.

11.2.4. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the `core` user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user `core`. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

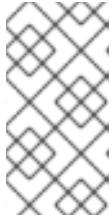
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

11.2.5. Obtaining the installation program

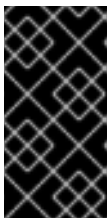
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.2.6. Adding Nutanix root CA certificates to your system trust

Because the installation program requires access to the Prism Central API, you must add your Nutanix trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the Prism Central web console, download the Nutanix root CA certificates.
2. Extract the compressed file that contains the Nutanix root CA certificates.
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

11.2.7. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Nutanix.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify that you have met the Nutanix networking requirements. For more information, see "Preparing to install on Nutanix".

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$. /openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

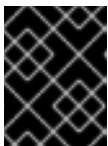
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **nutanix** as the platform to target.
 - iii. Enter the Prism Central domain name or IP address.
 - iv. Enter the port that is used to log into Prism Central.
 - v. Enter the credentials that are used to log into Prism Central.
The installation program connects to Prism Central.
 - vi. Select the Prism Element that will manage the OpenShift Container Platform cluster.
 - vii. Select the network subnet to use.
 - viii. Enter the virtual IP address that you configured for control plane API access.
 - ix. Enter the virtual IP address that you configured for cluster ingress.
 - x. Enter the base domain. This base domain must be the same one that you configured in the DNS records.
 - xi. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Optional: Update one or more of the default configuration parameters in the **install-config.yaml** file to customize the installation.
For more information about the parameters, see "Installation configuration parameters".
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

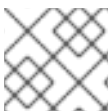


IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

11.2.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

11.2.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 11.3. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

11.2.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 11.4. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	<p>Object</p>  <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


11.2.7.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 11.5. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hypertreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 589 592 1395" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1442 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...

11.2.7.1.4. Additional Nutanix configuration parameters

Additional Nutanix configuration parameters are described in the following table:

Table 11.6. Additional Nutanix cluster parameters

Parameter	Description	Values
platform.nutanix.apiVIP	The virtual IP (VIP) address that you configured for control plane API access.	IP address
platform.nutanix.ingressVIP	The virtual IP (VIP) address that you configured for cluster ingress.	IP address

Parameter	Description	Values
platform.nutanix.pris mCentral.endpoint.a ddress	The Prism Central domain name or IP address.	String
platform.nutanix.pris mCentral.endpoint.p ort	The port that is used to log into Prism Central.	String
platform.nutanix.pris mCentral.password	The password for the Prism Central user name.	String
platform.nutanix.pris mCentral.username	The user name that is used to log into Prism Central.	String
platform.nutanix.pris mElements.endpoint. address	The Prism Element domain name or IP address. [1]	String
platform.nutanix.pris mElements.endpoint. port	The port that is used to log into Prism Element.	String
platform.nutanix.pris mElements.uuid	The universally unique identifier (UUID) for Prism Element.	String
platform.nutanix.sub netUUIDs	The UUID of the Prism Element network that contains the virtual IP addresses and DNS records that you configured. [2]	String
platform.nutanix.clu sterOSImage	Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server and pointing the installation program to the image.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <code>http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2</code>

1. The **prismElements** section holds a list of Prism Elements (clusters). A Prism Element encompasses all of the Nutanix resources, for example virtual machines and subnets, that are used to host the OpenShift Container Platform cluster. Only a single Prism Element is supported.
2. Only one subnet per OpenShift Container Platform cluster is supported.

11.2.7.2. Sample customized install-config.yaml file for Nutanix

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 3
  platform:
    nutanix: 4
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3
  platform:
    nutanix: 7
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
      osDisk:
        diskSizeGiB: 120
metadata:
  creationTimestamp: null
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
  - 172.30.0.0/16
platform:
  nutanix:
    apiVIP: 10.40.142.7 10
    ingressVIP: 10.40.142.8 11
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 12

```

```

port: 9440 13
password: <password> 14
username: <username> 15
prismElements:
- endpoint:
  address: your.prismelement.domainname
  port: 9440
  uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
subnetUUIDs:
- c7938dc6-7659-453e-a688-e26020c68e43
clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
16
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 17
fips: false 18
sshKey: ssh-ed25519 AAAA... 19

```

1 8 10 11 12 13 14 15 17 Required. The installation program prompts you for this value.

2 5 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3 6 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4 7 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

16 Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server and pointing the installation program to the image.

18 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

19

Optional: You can provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

11.2.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

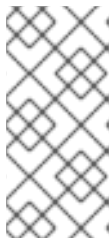
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

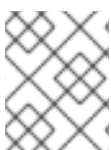
**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

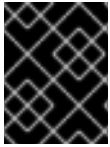
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

11.2.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

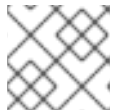
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

11.2.9. Configuring IAM for Nutanix

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an **install-config.yaml** file.

Procedure

1. Create a YAML file that contains the credentials data in the following format:

Credentials data format

```
credentials:
- type: basic_auth 1
  data:
    prismCentral: 2
      username: <username_for_prism_central>
      password: <password_for_prism_central>
    prismElements: 3
      - name: <name_of_prism_element>
        username: <username_for_prism_element>
        password: <password_for_prism_element>
```

- 1** Specify the authentication type. Only basic authentication is supported.
- 2** Specify the Prism Central credentials.
- 3** Optional: Specify the Prism Element credentials.

2. Set a **\$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
--from=$RELEASE_IMAGE \
--credentials-requests \
--cloud=nutanix \
--to=<path_to_directory_with_list_of_credentials_requests>/credrequests 1
```

- 1** Specify the path to the directory that contains the files for the component **CredentialsRequests** objects. If the specified directory does not exist, this command creates it.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
```



```
secretRef:
  name: nutanix-credentials
  namespace: openshift-machine-api
```

- If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on Nutanix

```
0000_30_machine-api-operator_00_credentials-request.yaml 1
```

- The Machine API Operator CR is required.

- Use the **ccoctl** tool to process all of the **CredentialsRequest** objects in the **credrequests** directory by running the following command:

```
$ ccoctl nutanix create-shared-secrets \
  --credentials-requests-dir=
  <path_to_directory_with_list_of_credentials_requests>/credrequests 1
  --output-dir=<ccoctl_output_dir> 2
  --credentials-source-filepath=<path_to_credentials_file> 3
```

- Specify the path to the directory that contains the files for the component **CredentialsRequests** objects.
- Specify the directory that contains the files of the component credentials secrets, under the **manifests** directory. By default, the **ccoctl** tool creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag.
- Optional: Specify the directory that contains the credentials data YAML file. By default, **ccoctl** expects this file to be in **<home_directory>/.**nutanix/credentials****. To specify a different directory, use the **--credentials-source-filepath** flag.

- Edit the **install-config.yaml** configuration file so that the **credentialsMode** parameter is set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
...
```

- Add this line to set the **credentialsMode** parameter to **Manual**.

- Create the installation manifests by running the following command:

```
$ openshift-install create manifests --dir <installation_directory> 1
```

- 1 Specify the path to the directory that contains the **install-config.yaml** file for your cluster.

8. Copy the generated credential files to the target manifests directory by running the following command:

```
$ cp <ccoctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

Verification

- Ensure that the appropriate secrets exist in the **manifests** directory.

```
$ ls ./<installation_directory>/manifests
```

Example output

```
total 64
-rw-r----- 1 <user> <user> 2335 Jul  8 12:22 cluster-config.yaml
-rw-r----- 1 <user> <user>  161 Jul  8 12:22 cluster-dns-02-config.yml
-rw-r----- 1 <user> <user>  864 Jul  8 12:22 cluster-infrastructure-02-config.yml
-rw-r----- 1 <user> <user>  191 Jul  8 12:22 cluster-ingress-02-config.yml
-rw-r----- 1 <user> <user> 9607 Jul  8 12:22 cluster-network-01-crd.yml
-rw-r----- 1 <user> <user>  272 Jul  8 12:22 cluster-network-02-config.yml
-rw-r----- 1 <user> <user>  142 Jul  8 12:22 cluster-proxy-01-config.yaml
-rw-r----- 1 <user> <user>  171 Jul  8 12:22 cluster-scheduler-02-config.yml
-rw-r----- 1 <user> <user>  200 Jul  8 12:22 cvo-overrides.yaml
-rw-r----- 1 <user> <user>  118 Jul  8 12:22 kube-cloud-config.yaml
-rw-r----- 1 <user> <user> 1304 Jul  8 12:22 kube-system-configmap-root-ca.yaml
-rw-r----- 1 <user> <user> 4090 Jul  8 12:22 machine-config-server-tls-secret.yaml
-rw-r----- 1 <user> <user> 3961 Jul  8 12:22 openshift-config-secret-pull-secret.yaml
-rw----- 1 <user> <user>  283 Jul  8 12:24 openshift-machine-api-nutanix-credentials-credentials.yaml
```

11.2.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

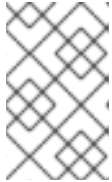
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



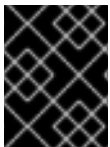
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

11.2.11. Configuring the default storage container

After you install the cluster, you must install the Nutanix CSI Operator and configure the default storage container for the cluster.

For more information, see the Nutanix documentation for [installing the CSI Operator](#) and [configuring registry storage](#).

11.2.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

11.2.13. Additional resources

- [About remote health monitoring](#)

11.2.14. Next steps

- [Opt out of remote health reporting](#)
- [Customize your cluster](#)

11.3. INSTALLING A CLUSTER ON NUTANIX IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.12, you can install a cluster on Nutanix infrastructure in a restricted network by creating an internal mirror of the installation release content.

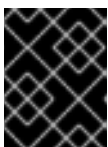
11.3.1. Prerequisites

- You have reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- The installation program requires access to port 9440 on Prism Central and Prism Element. You verified that port 9440 is accessible.
- If you use a firewall, you have met these prerequisites:
 - You confirmed that port 9440 is accessible. Control plane nodes must be able to reach Prism Central and Prism Element on port 9440 for the installation to succeed.
 - You configured the firewall to [grant access](#) to the sites that OpenShift Container Platform requires. This includes the use of Telemetry.
- If your Nutanix environment is using the default self-signed SSL/TLS certificate, replace it with a certificate that is signed by a CA. The installation program requires a valid CA-signed certificate to access to the Prism Central API. For more information about replacing the self-signed certificate, see the [Nutanix AOS Security Guide](#) .
If your Nutanix environment uses an internal CA to issue certificates, you must configure a cluster-wide proxy as part of the installation process. For more information, see [Configuring a custom PKI](#).

**IMPORTANT**

Use 2048-bit certificates. The installation fails if you use 4096-bit certificates with Prism Central 2022.x.

- You have a container image registry, such as Red Hat Quay. If you do not already have a registry, you can create a mirror registry using [mirror registry for Red Hat OpenShift](#) .
- You have used the [oc-mirror OpenShift CLI \(oc\) plugin](#) to mirror all of the required OpenShift Container Platform content and other images, including the Nutanix CSI Operator, to your mirror registry.

**IMPORTANT**

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

11.3.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

11.3.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

11.3.3. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

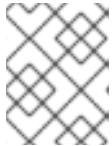
- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

11.3.4. Adding Nutanix root CA certificates to your system trust

Because the installation program requires access to the Prism Central API, you must add your Nutanix trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the Prism Central web console, download the Nutanix root CA certificates.
2. Extract the compressed file that contains the Nutanix root CA certificates.
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

11.3.5. Downloading the RHCOS cluster image

Prism Central requires access to the Red Hat Enterprise Linux CoreOS (RHCOS) image to install the cluster. You can use the installation program to locate and download the RHCOS image and make it available through an internal HTTP server or Nutanix Objects.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$. /openshift-install coreos print-stream-json
```

2. Use the output of the command to find the location of the Nutanix image, and click the link to download it.

Example output

```
"nutanix": {
  "release": "411.86.202210041459-0",
  "formats": {
    "qcow2": {
      "disk": {
        "location": "https://rhcos.mirror.openshift.com/art/storage/releases/rhcos-4.11/411.86.202210041459-0/x86_64/rhcos-411.86.202210041459-0-nutanix.x86_64.qcow2",
        "sha256":
          "42e227cac6f11ac37ee8a2f9528bb3665146566890577fd55f9b950949e5a54b"
```

3. Make the image available through an internal HTTP server or Nutanix Objects.

4. Note the location of the downloaded image. You update the **platform** section in the installation configuration file (**install-config.yaml**) with the image's location before deploying the cluster.

Snippet of an **install-config.yaml** file that specifies the RHCOS image

```
platform:
  nutanix:
    clusterOSImage: http://example.com/images/rhcos-411.86.202210041459-0-
    nutanix.x86_64.qcow2
```

11.3.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Nutanix.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSourcePolicy.yaml** file that was created when you mirrored your registry.
- Have the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image you download.
- Obtain the contents of the certificate for your mirror registry.
- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.
- Verify that you have met the Nutanix networking requirements. For more information, see "Preparing to install on Nutanix".

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them

into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **nutanix** as the platform to target.
 - iii. Enter the Prism Central domain name or IP address.
 - iv. Enter the port that is used to log into Prism Central.
 - v. Enter the credentials that are used to log into Prism Central.
The installation program connects to Prism Central.
 - vi. Select the Prism Element that will manage the OpenShift Container Platform cluster.
 - vii. Select the network subnet to use.
 - viii. Enter the virtual IP address that you configured for control plane API access.
 - ix. Enter the virtual IP address that you configured for cluster ingress.
 - x. Enter the base domain. This base domain must be the same one that you configured in the DNS records.
 - xi. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. In the **install-config.yaml** file, set the value of **platform.nutanix.clusterOSImage** to the image location or name. For example:

```
platform:
  nutanix:
    clusterOSImage: http://mirror.example.com/images/rhcos-47.83.202103221318-0-
    nutanix.x86_64.qcow2
```

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000":{"auth": "<credentials>","email":
  "you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

  /-----/
  -----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
  - mirrors:
    - <mirror_host_name>:5000/<repo_name>/release
      source: quay.io/openshift-release-dev/ocp-release
  - mirrors:
    - <mirror_host_name>:5000/<repo_name>/release
      source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSourcePolicy.yaml** file that was created when you mirrored the registry.

4. Optional: Update one or more of the default configuration parameters in the **install.config.yaml** file to customize the installation.
For more information about the parameters, see "Installation configuration parameters".
5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

11.3.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

11.3.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 11.7. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

11.3.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 11.8. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


11.3.6.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 11.9. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hypertreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div style="margin-left: 10px;"> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 586 592 1391" style="background-color: black; width: 65px; height: 359px; margin-bottom: 10px;"></div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="488 1440 592 1637" style="background-color: black; width: 65px; height: 88px; margin-bottom: 10px;"></div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

11.3.6.1.4. Additional Nutanix configuration parameters

Additional Nutanix configuration parameters are described in the following table:

Table 11.10. Additional Nutanix cluster parameters

Parameter	Description	Values
platform.nutanix.apiVIP	The virtual IP (VIP) address that you configured for control plane API access.	IP address
platform.nutanix.ingressVIP	The virtual IP (VIP) address that you configured for cluster ingress.	IP address

Parameter	Description	Values
platform.nutanix.prismCentral.endpoint.address	The Prism Central domain name or IP address.	String
platform.nutanix.prismCentral.endpoint.port	The port that is used to log into Prism Central.	String
platform.nutanix.prismCentral.password	The password for the Prism Central user name.	String
platform.nutanix.prismCentral.username	The user name that is used to log into Prism Central.	String
platform.nutanix.prismElements.endpoint.address	The Prism Element domain name or IP address. [1]	String
platform.nutanix.prismElements.endpoint.port	The port that is used to log into Prism Element.	String
platform.nutanix.prismElements.uuid	The universally unique identifier (UUID) for Prism Element.	String
platform.nutanix.subnetUUIDs	The UUID of the Prism Element network that contains the virtual IP addresses and DNS records that you configured. [2]	String
platform.nutanix.clusterOSImage	Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server and pointing the installation program to the image.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <code>http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2</code>

1. The **prismElements** section holds a list of Prism Elements (clusters). A Prism Element encompasses all of the Nutanix resources, for example virtual machines and subnets, that are used to host the OpenShift Container Platform cluster. Only a single Prism Element is supported.
2. Only one subnet per OpenShift Container Platform cluster is supported.

11.3.6.2. Sample customized install-config.yaml file for Nutanix

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.



IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 3
  platform:
    nutanix: 4
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3
  platform:
    nutanix: 7
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
      osDisk:
        diskSizeGiB: 120
metadata:
  creationTimestamp: null
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
  - 172.30.0.0/16
platform:
  nutanix:
    apiVIP: 10.40.142.7 10
    ingressVIP: 10.40.142.8 11
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 12

```

```

port: 9440 13
password: <password> 14
username: <username> 15
prismElements:
- endpoint:
  address: your.prismelement.domainname
  port: 9440
  uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
subnetUUIDs:
- c7938dc6-7659-453e-a688-e26020c68e43
clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
16
credentialsMode: Manual
publish: External
pullSecret: '{"auths":{"<local_registry>":{"auth": "<credentials>","email": "you@example.com"}}}' 17
fips: false 18
sshKey: ssh-ed25519 AAAA... 19
additionalTrustBundle: | 20
----BEGIN CERTIFICATE-----
////////////////////
----END CERTIFICATE----
imageContentSources: 21
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev

```

1 8 10 11 12 13 14 15 Required. The installation program prompts you for this value.

2 5 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3 6 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.



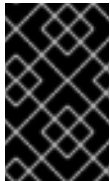
IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4 7 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

9 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.

- 16 Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the
- 17 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 18 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS Validated or Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 19 Optional: You can provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 20 Provide the contents of the certificate file that you used for your mirror registry.
- 21 Provide these values from the **metadata.name: release-0** section of the **imageContentSourcePolicy.yaml** file that was created when you mirrored the registry.

11.3.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

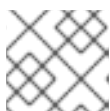
For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

11.3.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

11.3.8. Configuring IAM for Nutanix

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an **install-config.yaml** file.

Procedure

1. Create a YAML file that contains the credentials data in the following format:

Credentials data format

```
credentials:
- type: basic_auth 1
  data:
    prismCentral: 2
      username: <username_for_prism_central>
      password: <password_for_prism_central>
    prismElements: 3
      - name: <name_of_prism_element>
        username: <username_for_prism_element>
        password: <password_for_prism_element>
```

- 1 Specify the authentication type. Only basic authentication is supported.
- 2 Specify the Prism Central credentials.
- 3 Optional: Specify the Prism Element credentials.

2. Set a **\$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk 'release image/ {print $3}')
```

3. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --cloud=nutanix \
  --to=<path_to_directory_with_list_of_credentials_requests>/credrequests 1
```

- 1 Specify the path to the directory that contains the files for the component **CredentialsRequests** objects. If the specified directory does not exist, this command creates it.

Sample CredentialsRequest object

```

apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
  secretRef:
    name: nutanix-credentials
    namespace: openshift-machine-api

```

4. If your cluster uses cluster capabilities to disable one or more optional components, delete the **CredentialsRequest** custom resources for any disabled components.

Example credrequests directory contents for OpenShift Container Platform 4.12 on Nutanix

```
0000_30_machine-api-operator_00_credentials-request.yaml 1
```

- 1 The Machine API Operator CR is required.
5. Use the **ccoctl** tool to process all of the **CredentialsRequest** objects in the **credrequests** directory by running the following command:

```

$ ccoctl nutanix create-shared-secrets \
--credentials-requests-dir=
<path_to_directory_with_list_of_credentials_requests>/credrequests 1
--output-dir=<ccoctl_output_dir> 2
--credentials-source-filepath=<path_to_credentials_file> 3

```

- 1 Specify the path to the directory that contains the files for the component **CredentialsRequests** objects.
- 2 Specify the directory that contains the files of the component credentials secrets, under the **manifests** directory. By default, the **ccoctl** tool creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag.
- 3 Optional: Specify the directory that contains the credentials data YAML file. By default, **ccoctl** expects this file to be in **<home_directory>/./nutanix/credentials**. To specify a different directory, use the **--credentials-source-filepath** flag.

6. Edit the **install-config.yaml** configuration file so that the **credentialsMode** parameter is set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
...
```

- 1** Add this line to set the **credentialsMode** parameter to **Manual**.

7. Create the installation manifests by running the following command:

```
$ openshift-install create manifests --dir <installation_directory> 1
```

- 1** Specify the path to the directory that contains the **install-config.yaml** file for your cluster.

8. Copy the generated credential files to the target manifests directory by running the following command:

```
$ cp <cocctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

Verification

- Ensure that the appropriate secrets exist in the **manifests** directory.

```
$ ls ./<installation_directory>/manifests
```

Example output

```
total 64
-rw-r----- 1 <user> <user> 2335 Jul  8 12:22 cluster-config.yaml
-rw-r----- 1 <user> <user>  161 Jul  8 12:22 cluster-dns-02-config.yml
-rw-r----- 1 <user> <user>  864 Jul  8 12:22 cluster-infrastructure-02-config.yml
-rw-r----- 1 <user> <user>  191 Jul  8 12:22 cluster-ingress-02-config.yml
-rw-r----- 1 <user> <user> 9607 Jul  8 12:22 cluster-network-01-crd.yml
-rw-r----- 1 <user> <user>  272 Jul  8 12:22 cluster-network-02-config.yml
-rw-r----- 1 <user> <user>  142 Jul  8 12:22 cluster-proxy-01-config.yaml
-rw-r----- 1 <user> <user>  171 Jul  8 12:22 cluster-scheduler-02-config.yml
-rw-r----- 1 <user> <user>  200 Jul  8 12:22 cvo-overrides.yaml
-rw-r----- 1 <user> <user>  118 Jul  8 12:22 kube-cloud-config.yaml
-rw-r----- 1 <user> <user> 1304 Jul  8 12:22 kube-system-configmap-root-ca.yaml
-rw-r----- 1 <user> <user> 4090 Jul  8 12:22 machine-config-server-tls-secret.yaml
-rw-r----- 1 <user> <user> 3961 Jul  8 12:22 openshift-config-secret-pull-secret.yaml
-rw----- 1 <user> <user>  283 Jul  8 12:24 openshift-machine-api-nutanix-credentials-credentials.yaml
```

11.3.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
```

```
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

11.3.10. Post installation

Complete the following steps to complete the configuration of your cluster.

11.3.10.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

11.3.10.2. Installing the policy resources into the cluster

Mirroring the OpenShift Container Platform content using the oc-mirror OpenShift CLI (oc) plugin creates resources, which include **catalogSource-certified-operator-index.yaml** and **imageContentSourcePolicy.yaml**.

- The **ImageContentSourcePolicy** resource associates the mirror registry with the source registry and redirects image pull requests from the online registries to the mirror registry.
- The **CatalogSource** resource is used by Operator Lifecycle Manager (OLM) to retrieve information about the available Operators in the mirror registry, which lets users discover and install Operators.

After you install the cluster, you must install these resources into the cluster.

Prerequisites

- You have mirrored the image set to the registry mirror in the disconnected environment.
- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Log in to the OpenShift CLI as a user with the **cluster-admin** role.
2. Apply the YAML files from the results directory to the cluster:

```
$ oc apply -f ./oc-mirror-workspace/results-<id>/
```

Verification

1. Verify that the **ImageContentSourcePolicy** resources were successfully installed:

```
$ oc get imagecontentsourcepolicy
```

2. Verify that the **CatalogSource** resources were successfully installed:

```
$ oc get catalogsource --all-namespaces
```

11.3.10.3. Configuring the default storage container

After you install the cluster, you must install the Nutanix CSI Operator and configure the default storage container for the cluster.

For more information, see the Nutanix documentation for [installing the CSI Operator](#) and [configuring registry storage](#).

11.3.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

11.3.12. Additional resources

- [About remote health monitoring](#)

11.3.13. Next steps

- If necessary, see [Opt out of remote health reporting](#)
- If necessary, see [Registering your disconnected cluster](#)
- [Customize your cluster](#)

11.4. UNINSTALLING A CLUSTER ON NUTANIX

You can remove a cluster that you deployed to Nutanix.

11.4.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

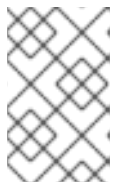
Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

- Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 12. INSTALLING ON BARE METAL

12.1. PREPARING FOR BARE METAL CLUSTER INSTALLATION

12.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You have read the documentation on [selecting a cluster installation method and preparing it for users](#).

12.1.2. Planning a bare metal cluster for OpenShift Virtualization

If you will use OpenShift Virtualization, it is important to be aware of several requirements before you install your bare metal cluster.

- If you want to use live migration features, you must have multiple worker nodes *at the time of cluster installation*. This is because live migration requires the cluster-level high availability (HA) flag to be set to true. The HA flag is set when a cluster is installed and cannot be changed afterwards. If there are fewer than two worker nodes defined when you install your cluster, the HA flag is set to false for the life of the cluster.



NOTE

You can install OpenShift Virtualization on a single-node cluster, but single-node OpenShift does not support high availability.

- Live migration requires shared storage. Storage for OpenShift Virtualization must support and use the ReadWriteMany (RWX) access mode.
- If you plan to use Single Root I/O Virtualization (SR-IOV), ensure that your network interface controllers (NICs) are supported by OpenShift Container Platform.

Additional resources

- [Preparing your cluster for OpenShift Virtualization](#)
- [Getting started with OpenShift Virtualization](#)
- [About Single Root I/O Virtualization \(SR-IOV\) hardware networks](#)
- [Connecting a virtual machine to an SR-IOV network](#)

12.1.3. Choosing a method to install OpenShift Container Platform on bare metal

The OpenShift Container Platform installation program offers four methods for deploying a cluster:

- **Interactive:** You can deploy a cluster with the web-based [Assisted Installer](#). This is the recommended approach for clusters with networks connected to the internet. The Assisted Installer is the easiest way to install OpenShift Container Platform, it provides smart defaults, and it performs pre-flight validations before installing the cluster. It also provides a RESTful API for automation and advanced configuration scenarios.

- **Local Agent-based:** You can deploy a cluster locally with the [agent-based installer](#) for air-gapped or restricted networks. It provides many of the benefits of the Assisted Installer, but you must download and configure the [agent-based installer](#) first. Configuration is done with a commandline interface. This approach is ideal for air-gapped or restricted networks.
- **Automated:** You can [deploy a cluster on installer-provisioned infrastructure](#) and the cluster it maintains. The installer uses each cluster host's baseboard management controller (BMC) for provisioning. You can deploy clusters with both connected or air-gapped or restricted networks.
- **Full control:** You can deploy a cluster on [infrastructure that you prepare and maintain](#), which provides maximum customizability. You can deploy clusters with both connected or air-gapped or restricted networks.

The clusters have the following characteristics:

- Highly available infrastructure with no single points of failure is available by default.
- Administrators maintain control over what updates are applied and when.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

12.1.3.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on bare metal infrastructure that is provisioned by the OpenShift Container Platform installation program, by using the following method:

- [Installing an installer-provisioned cluster on bare metal](#) You can install OpenShift Container Platform on bare metal by using installer provisioning.

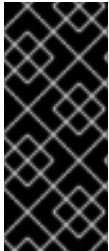
12.1.3.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on bare metal infrastructure that you provision, by using one of the following methods:

- [Installing a user-provisioned cluster on bare metal](#) You can install OpenShift Container Platform on bare metal infrastructure that you provision. For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.
- [Installing a user-provisioned bare metal cluster with network customizations](#) You can install a bare metal cluster on user-provisioned infrastructure with network-customizations. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. Most of the network customizations must be applied at the installation stage.
- [Installing a user-provisioned bare metal cluster on a restricted network](#) You can install a user-provisioned bare metal cluster on a restricted or disconnected network by using a mirror registry. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

12.2. INSTALLING A USER-PROVISIONED CLUSTER ON BARE METAL

In OpenShift Container Platform 4.12, you can install a cluster on bare metal infrastructure that you provision.

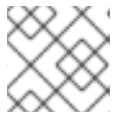


IMPORTANT

While you might be able to follow this procedure to deploy a cluster on virtualized or cloud environments, you must be aware of additional considerations for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you attempt to install an OpenShift Container Platform cluster in such an environment.

12.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

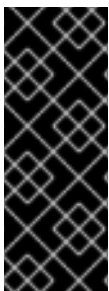
Be sure to also review this site list if you are configuring a proxy.

12.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

Additional resources

- See [Installing a user-provisioned bare metal cluster on a restricted network](#) for more information about performing a restricted network installation on bare metal infrastructure that you provision.

12.2.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

12.2.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

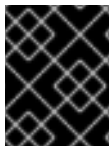
Table 12.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



NOTE

As an exception, you can run zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production. Running one compute machine is not supported.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

12.2.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 12.2. Minimum resource requirements

Machine	Operating System	CPU [1]	RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

12.2.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- See [Configuring a three-node cluster](#) for details about deploying three-node clusters in bare metal environments.
- See [Approving the certificate signing requests for your machines](#) for more information about approving cluster certificate signing requests after installation.

12.2.3.4. Requirements for baremetal clusters on vSphere

Ensure you enable the **disk.EnableUUID** parameter on all virtual machines in your cluster.

Additional resources

- See [Installing RHCOS and starting the OpenShift Container Platform bootstrap process](#) for details on setting the **disk.EnableUUID** parameter's value to **TRUE** on VMware vSphere for user-provisioned infrastructure.

12.2.3.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

12.2.3.5.1. Setting the cluster node hostnames through DHCP

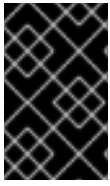
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

12.2.3.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 12.3. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 12.4. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 12.5. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

12.2.3.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS

record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 12.6. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify `etcd` host and `SRV` records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

12.2.3.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 12.1. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
```

```
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 12.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
```

```

;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF

```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

Additional resources

- [Validating DNS resolution for user-provisioned infrastructure](#)

12.2.3.7. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

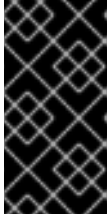


NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 12.7. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer:** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 12.8. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

12.2.3.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running `setsebool -P haproxy_connect_any=1`.

Example 12.3. Sample API and application Ingress load balancer configuration

```
global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request 10s
```

```

timeout queue      1m
timeout connect   10s
timeout client    1m
timeout server    1m
timeout http-keep-alive 10s
timeout check     10s
maxconn          3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
    server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
    fall 2 rise 3
    server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
    fall 2 rise 3
    server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
    fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:443 check inter 1s
  server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:80 check inter 1s
  server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

12.2.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.

**NOTE**

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

Additional resources

- [Requirements for a cluster with user-provisioned infrastructure](#)
- [Installing RHCOS and starting the OpenShift Container Platform bootstrap process](#)
- [Setting the cluster node hostnames through DHCP](#)
- [Advanced RHCOS installation configuration](#)
- [Networking requirements for user-provisioned infrastructure](#)
- [User-provisioned DNS requirements](#)
- [Validating DNS resolution for user-provisioned infrastructure](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.2.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.

- a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1** Provides the record name for the Kubernetes internal API.
- 2** Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

Additional resources

- [User-provisioned DNS requirements](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.2.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

■


```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

Additional resources

- [Verifying node health](#)

12.2.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

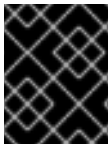
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

12.2.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.

To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

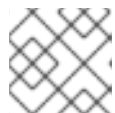
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

12.2.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

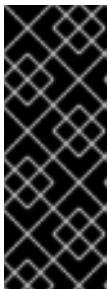
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

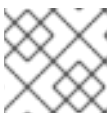
```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

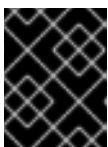
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

12.2.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

12.2.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 12.9. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

12.2.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.

```
networking:
  clusterNetwork:
```


- cidr: 10.128.0.0/14
hostPrefix: 23
- cidr: fd00:10:128::/56
hostPrefix: 64
- serviceNetwork:
 - 172.30.0.0/16
 - fd00:172:16::/112

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 12.10. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>
networking.network Type	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	<p>The IP address blocks for pods.</p> <p>The default value is 10.128.0.0/14 with a host prefix of /23.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre style="margin-left: 20px;">networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23 - cidr: fd01::/48 hostPrefix: 64</pre>


Parameter	Description	Values
networking.clusterNetwork.cidr	<p>Required if you use networking.clusterNetwork. An IP address block.</p> <p>If you use the OpenShift SDN network plugin, specify an IPv4 network. If you use the OVN-Kubernetes network plugin, you can specify IPv4 and IPv6 networks.</p>	<p>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32. The prefix length for an IPv6 block is between 0 and 128. For example, 10.128.0.0/14 or fd01::/48.</p>
networking.clusterNetwork.hostPrefix	<p>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides $2^{(32 - 23) - 2}$ pod IP addresses.</p>	<p>A subnet prefix.</p> <p>For an IPv4 network the default value is 23. For an IPv6 network the default value is 64. The default value is also the minimum value for IPv6.</p>
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p> <p>If you use the OVN-Kubernetes network plugin, you can specify an IP address block for both of the IPv4 and IPv6 address families.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16 - fd02::/112</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16 or fd00::/48.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


12.2.9.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 12.11. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String

Parameter	Description	Values
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.

Parameter	Description	Values
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 20px;"> <div style="width: 20px; height: 50px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	<p>For example, sshKey: ssh-ed25519 AAAA...</p>

12.2.9.2. Sample install-config.yaml file for bare metal

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com ①
compute: ②
- hyperthreading: Enabled ③
  name: worker
  replicas: 0 ④
controlPlane: ⑤
  hyperthreading: Enabled ⑥
  name: master
  replicas: 3 ⑦
metadata:
  name: test ⑧

```

```

networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14 9
      hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
    - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.

- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for your platform.



IMPORTANT

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

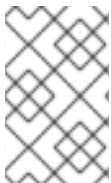
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

Additional resources

- See [Load balancing requirements for user-provisioned infrastructure](#) for more information on the API and application ingress load balancing requirements.
- See [Enabling cluster capabilities](#) for more information on enabling cluster capabilities that were disabled prior to installation.
- See [Optional cluster capabilities in OpenShift Container Platform 4.12](#) for more information about the features provided by each capability.

12.2.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

**NOTE**

For bare metal installations, if you do not assign node IP addresses from the range that is specified in the **networking.machineNetwork[].cidr** field in the **install-config.yaml** file, you must include them in the **proxy.noProxy** field.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

12.2.9.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

**NOTE**

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

12.2.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.



WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.



IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

Additional resources

- See [Recovering from expired control plane certificates](#) for more information about recovering kubelet certificates.

12.2.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.

**NOTE**

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Only RHEL 8 compute machines are supported.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments:** You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the **APPEND** parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special **coreos.inst.*** arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.
- **Ignition configs:** OpenShift Container Platform Ignition config files (*.ign) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the **coreos-installer** to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.
- **coreos-installer:** You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the **coreos-installer** command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.

**NOTE**

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

12.2.11.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.

- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

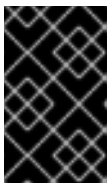
Procedure

1. Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

2. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

3. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left     Speed
  0   0   0   0   0   0   0   0   0  --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

4. Although it is possible to obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS images are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep '\.iso[^\.]'
```

Example output

```
"location": "<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-
```

```
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live.x86_64.iso",
```



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:
 - Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.



NOTE

It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

- 1** **1** You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.
- 2** The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.



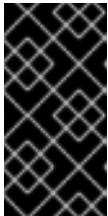
NOTE

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the `/dev/sda` device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

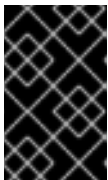
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.2.11.2. Installing RHCOS by using PXE or iPXE booting

You can use PXE or iPXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE or iPXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left   Speed
  0   0   0    0    0    0    0    0  0  --:--:--  --:--:--  --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

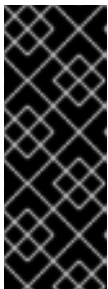
3. Although it is possible to obtain the RHCOS **kernel**, **initramfs** and **rootfs** files that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS files are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep -Eo "https.*(kernel-|initramfs.|rootfs.)w+(\.img)?"
```

Example output

■

```
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



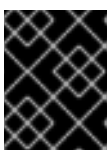
IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** rhcos-<version>-live-kernel-<architecture>
- **initramfs:** rhcos-<version>-live-initramfs.<architecture>.img
- **rootfs:** rhcos-<version>-live-rootfs.<architecture>.img

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.
6. Configure PXE or iPXE installation for the RHCOS images and begin the installation.

Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE (**x86_64**):

```

DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3

```

- 1 1 Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

- For iPXE (**x86_64 + aarch64**):

```

kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot

```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **kernel** parameter value is the location of the **kernel** file, the **initrd=main** argument is needed for booting on UEFI systems, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file.

- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your HTTP server.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **kernel** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.



NOTE

To network boot the CoreOS **kernel** on **aarch64** architecture, you need to use a version of iPXE build with the **IMAGE_GZIP** option enabled. See [IMAGE_GZIP option in iPXE](#).

- For PXE (with UEFI and Grub as second stage) on **aarch64**:

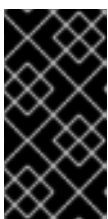
```

menuentry 'Install CoreOS' {
  linux rhcos-<version>-live-kernel-<architecture>
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
  <architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
  initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}

```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP/TFTP server. The **kernel** parameter value is the location of the **kernel** file on your TFTP server. The **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file on your HTTP Server.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your TFTP server.

7. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.
9. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. Continue to create the machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.2.11.3. Advanced RHCOS installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
- Customizing a live ISO or PXE boot image

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

12.2.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

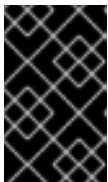
- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

Procedure

1. Boot the ISO installer.
2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as **nmcli** or **nmtui**.
3. Run the **coreos-installer** command to install the system, adding the **--copy-network** option to copy networking configuration. For example:

```
$ sudo coreos-installer install --copy-network \
  --ignition-url=http://host/worker.ign /dev/sda
```



IMPORTANT

The **--copy-network** option only copies networking configuration found under **/etc/NetworkManager/system-connections**. In particular, it does not copy the system hostname.

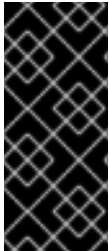
4. Reboot into the installed system.

Additional resources

- See [Getting started with nmcli](#) and [Getting started with nmtui](#) in the RHEL 8 documentation for more information about the **nmcli** and **nmtui** tools.

12.2.11.3.2. Disk partitioning

Disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless you override the default partitioning configuration. During the RHCOS installation, the size of the root file system is increased to use any remaining available space on the target device.



IMPORTANT

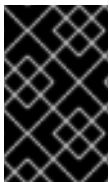
The use of a custom partition scheme on your node might result in OpenShift Container Platform not monitoring or alerting on some node partitions. If you override the default partitioning, see [Understanding OpenShift File System Monitoring \(eviction conditions\)](#) for more information about how OpenShift Container Platform monitors your host file systems.

OpenShift Container Platform monitors the following two filesystem identifiers:

- **nodefs**, which is the filesystem that contains **/var/lib/kubelet**
- **imagefs**, which is the filesystem that contains **/var/lib/containers**

For the default partition scheme, **nodefs** and **imagefs** monitor the same root filesystem, **/**.

To override the default partitioning when installing RHCOS on an OpenShift Container Platform cluster node, you must create separate partitions.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.

Consider a situation where you want to add a separate storage partition for your containers and container images. For example, by mounting **/var/lib/containers** in a separate partition, the kubelet separately monitors **/var/lib/containers** as the **imagefs** directory and the root file system as the **nodefs** directory.



IMPORTANT

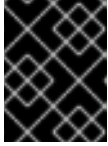
If you have resized your disk size to host a larger file system, consider creating a separate **/var/lib/containers** partition. Consider resizing a disk that has an **xfs** format to reduce CPU time issues caused by a high number of allocation groups.

12.2.11.3.2.1. Creating a separate **/var** partition

In general, you should use the default disk partitioning that is created during the RHCOS installation. However, there are cases where you might want to create a separate partition for a directory that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** directory or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate `/var` partition.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

The use of a separate partition for the `/var` directory or a subdirectory of `/var` also prevents data growth in the partitioned directory from filling up the root file system.

The following procedure sets up a separate `/var` partition by adding a machine config manifest that is wrapped into the Ignition config file for a node type during the preparation phase of an installation.

Procedure

1. On your installation host, change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ openshift-install create manifests --dir <installation_directory>
```

2. Create a Butane config that configures the additional partition. For example, name the file `$HOME/clusterconfig/98-var-partition.bu`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum offset value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no offset value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future

reinstalls of RHCOS might overwrite the beginning of the data partition.

- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for compute nodes, if the different instance types do not have the same device name.

3. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. Create the Ignition config files:

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

The files in the **<installation_directory>/manifest** and **<installation_directory>/openshift** directories are wrapped into the Ignition config files, including the file that contains the **98-var-partition** custom **MachineConfig** object.

Next steps

- You can apply the custom disk partitioning by referencing the Ignition config files during the RHCOS installations.

12.2.11.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the **coreos-installer** command that cause the installer to maintain one or more existing partitions. For a PXE installation, you can add **coreos.inst.*** options to the **APPEND** parameter to preserve partitions.

Saved partitions might be data partitions from an existing OpenShift Container Platform system. You can identify the disk partitions you want to keep either by partition label or by number.



NOTE

If you save existing partitions, and those partitions do not leave enough space for RHCOS, the installation will fail without damaging the saved partitions.

Retaining existing partitions during an ISO installation

This example preserves any partition in which the partition label begins with **data** (**data***):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the **coreos-installer** in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
  --save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, **coreos-installer** recreates the partition immediately.

Retaining existing partitions during a PXE installation

This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

This **APPEND** option preserves partition 6:

```
coreos.inst.save_partindex=6
```

12.2.11.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config:** Every manual RHCOS installation needs to pass one of the Ignition config files generated by **openshift-installer**, such as **bootstrap.ign**, **master.ign** and **worker.ign**, to carry out the installation.



IMPORTANT

It is not recommended to modify these Ignition config files directly. You can update the manifest files that are wrapped into the Ignition config files, as outlined in examples in the preceding sections.

For PXE installations, you pass the Ignition configs on the **APPEND** line using the **coreos.inst.ignition_url=** option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the **coreos-installer** command line with the **--ignition-url=** option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config:** This type can be created by using the **coreos-installer customize** subcommand and its various options. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be done once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

For PXE or ISO boots, you can create the Ignition config and **APPEND** the **ignition.config.url=** option to identify the location of the Ignition config. You also need to append **ignition.firstboot** or **ignition.platform.id=metal** or the **ignition.config.url** option will be ignored.

12.2.11.3.4. Default console configuration

Red Hat Enterprise Linux CoreOS (RHCOS) nodes installed from an OpenShift Container Platform 4.12 boot image use a default console that is meant to accommodate most virtualized and bare metal setups. Different cloud and virtualization platforms may use different default settings depending on the chosen architecture. Bare metal installations use the kernel default settings which typically means the graphical console is the primary console and the serial console is disabled.

The default consoles may not match your specific hardware configuration or you might have specific needs that require you to adjust the default console. For example:

- You want to access the emergency shell on the console for debugging purposes.
- Your cloud platform does not provide interactive access to the graphical console, but provides a serial console.
- You want to enable multiple consoles.

Console configuration is inherited from the boot image. This means that new nodes in existing clusters are unaffected by changes to the default console.

You can configure the console for bare metal installations in the following ways:

- Using **coreos-installer** manually on the command line.
- Using the **coreos-installer iso customize** or **coreos-installer pxe customize** subcommands with the **--dest-console** option to create a custom image that automates the process.



NOTE

For advanced customization, perform console configuration using the **coreos-installer iso** or **coreos-installer pxe** subcommands, and not kernel arguments.

12.2.11.3.5. Enabling the serial console for PXE and ISO installations

By default, the Red Hat Enterprise Linux CoreOS (RHCOS) serial console is disabled and all output is written to the graphical console. You can enable the serial console for an ISO installation and reconfigure the bootloader so that output is sent to both the serial console and the graphical console.

Procedure

1. Boot the ISO installer.
2. Run the **coreos-installer** command to install the system, adding the **--console** option once to specify the graphical console, and a second time to specify the serial console:

```
$ coreos-installer install \
  --console=tty0 \ 1
  --console=ttyS0,<options> \ 2
  --ignition-url=http://host/worker.ign /dev/sda
```

- 1 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 2 The desired primary console. In this case the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **11520n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see [Linux kernel serial console](#) documentation.

3. Reboot into the installed system.



NOTE

A similar outcome can be obtained by using the **coreos-installer install --append-karg** option, and specifying the console with **console=**. However, this will only set the console for the kernel and not the bootloader.

To configure a PXE installation, make sure the **coreos.inst.install_dev** kernel command line option is omitted, and use the shell prompt to run **coreos-installer** manually using the above ISO installation procedure.

12.2.11.3.6. Customizing a live RHCOS ISO or PXE install

You can use the live ISO image or PXE environment to install RHCOS by injecting an Ignition config file directly into the image. This creates a customized image that you can use to provision your system.

For an ISO image, the mechanism to do this is the **coreos-installer iso customize** subcommand, which modifies the **.iso** file with your configuration. Similarly, the mechanism for a PXE environment is the **coreos-installer pxe customize** subcommand, which creates a new **initramfs** file that includes your customizations.

The **customize** subcommand is a general purpose tool that can embed other types of customizations as well. The following tasks are examples of some of the more common customizations:

- Inject custom CA certificates for when corporate security policy requires their use.
- Configure network settings without the need for kernel arguments.
- Embed arbitrary preinstall and post-install scripts or binaries.

12.2.11.3.7. Customizing a live RHCOS ISO image

You can customize a live RHCOS ISO image directly with the **coreos-installer iso customize** subcommand. When you boot the ISO image, the customizations are applied automatically.

You can use this feature to configure the ISO image to automatically install RHCOS.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to inject the Ignition config directly into the ISO image:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda 2
```

- 1 The Ignition config file that is generated from the **openshift-installer** installation program.
- 2 When you specify this option, the ISO image automatically runs an installation. Otherwise, the image remains configured for installation, but does not install automatically unless you specify the **coreos.inst.install_dev** kernel argument.

3. Optional: To remove the ISO image customizations and return the image to its pristine state, run:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now re-customize the live ISO image or use it in its pristine state.

Applying your customizations affects every subsequent boot of RHCOS.

12.2.11.3.7.1. Modifying a live install ISO image to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

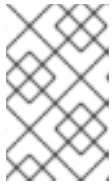
Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image to enable the serial console to receive output:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/sda 4
```

- 1 The location of the Ignition config to install.

- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. In this case, **/dev/sda**. If you omit this option, the ISO image automatically runs the installation program which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.



NOTE

The **--dest-console** option affects the installed system and not the live ISO system. To modify the console for a live ISO system, use the **--live-karg-append** option and specify the console with **console=**.

Your customizations are applied and affect every subsequent boot of the ISO image.

3. Optional: To remove the ISO image customizations and return the image to its original state, run the following command:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now recustomize the live ISO image or use it in its original state.

12.2.11.3.7.2. Modifying a live install ISO image to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



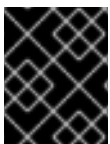
NOTE

Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image for use with a custom CA:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```



IMPORTANT

The **coreos.inst.ignition_url** kernel parameter does not work with the **--ignition-ca** flag. You must use the **--dest-ignition** flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.2.11.3.7.3. Modifying a live install ISO image with customized network settings

You can embed a NetworkManager keyfile into the live ISO image and pass it through to the installed system with the **--network-keyfile** flag of the **customize** subcommand.



WARNING

When creating a connection profile, you must use a **.nmconnection** filename extension in the filename of the connection profile. If you do not use a **.nmconnection** filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Create a connection profile for a bonded interface. For example, create the **bond0.nmconnection** file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
```



```
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image with your configured networking:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection
```

Network settings are applied to the live system and are carried over to the destination system.

12.2.11.3.8. Customizing a live RHCOS PXE environment

You can customize a live RHCOS PXE environment directly with the **coreos-installer pxe customize** subcommand. When you boot the PXE environment, the customizations are applied automatically.

You can use this feature to configure the PXE environment to automatically install RHCOS.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new **initramfs** file that contains the customizations from your Ignition config:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3
```

- 1 The Ignition config file that is generated from **openshift-installer**.
- 2 When you specify this option, the PXE environment automatically runs an install. Otherwise, the image remains configured for installing, but does not do so automatically unless you specify the **coreos.inst.install_dev** kernel argument.
- 3 Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Applying your customizations affects every subsequent boot of RHCOS.

12.2.11.3.8.1. Modifying a live install PXE environment to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new customized **initramfs** file that enables the serial console to receive output:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--dest-ignition <path> \ 1
--dest-console tty0 \ 2
--dest-console ttyS0,<options> \ 3
--dest-device /dev/sda \ 4
-o rhcos-<version>-custom-initramfs.x86_64.img 5
```

- 1 The location of the Ignition config to install.
- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. If you omit this option, the PXE environment automatically runs the installer which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.
- 5 Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Your customizations are applied and affect every subsequent boot of the PXE environment.

12.2.11.3.8.2. Modifying a live install PXE environment to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



NOTE

Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file for use with a custom CA:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --ignition-ca cert.pem \
  -o rhcos-<version>-custom-initramfs.x86_64.img
```

3. Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.



IMPORTANT

The **coreos.inst.ignition_url** kernel parameter does not work with the **--ignition-ca** flag. You must use the **--dest-ignition** flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.2.11.3.8.3. Modifying a live install PXE environment with customized network settings

You can embed a NetworkManager keyfile into the live PXE environment and pass it through to the installed system with the **--network-keyfile** flag of the **customize** subcommand.



WARNING

When creating a connection profile, you must use a **.nmconnection** filename extension in the filename of the connection profile. If you do not use a **.nmconnection** filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Create a connection profile for a bonded interface. For example, create the **bond0.nmconnection** file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file that contains your configured networking:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--network-keyfile bond0.nmconnection \
--network-keyfile bond0-proxy-em1.nmconnection \
--network-keyfile bond0-proxy-em2.nmconnection \
-o rhcos-<version>-custom-initramfs.x86_64.img
```

- Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present. Network settings are applied to the live system and are carried over to the destination system.

12.2.11.3.9. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

12.2.11.3.9.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**

- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

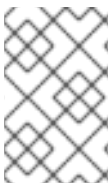
Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option. Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

12.2.11.3.9.2. coreos-installer options for ISO and PXE installations

You can install RHCOS by running **coreos-installer install <options> <device>** at the command prompt, after booting into the RHCOS live environment from an ISO image.

The following table shows the subcommands, options, and arguments you can pass to the **coreos-installer** command.


Table 12.12. coreos-installer subcommands, command-line options, and arguments

coreos-installer install subcommand

<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer install <options> <device>	Embed an Ignition config in an ISO image.
coreos-installer install subcommand options	
<i>Option</i>	<i>Description</i>
-u, --image-url <url>	Specify the image URL manually.
-f, --image-file <path>	Specify a local image file manually. Used for debugging.
-i, --ignition-file <path>	Embed an Ignition config from a file.
-l, --ignition-url <URL>	Embed an Ignition config from a URL.
--ignition-hash <digest>	Digest type-value of the Ignition config.
-p, --platform <name>	Override the Ignition platform ID for the installed system.
--console <spec>	Set the kernel and bootloader console for the installed system. For more information about the format of <spec> , see the Linux kernel serial console documentation.
--append-karg <arg>...	Append a default kernel argument to the installed system.
--delete-karg <arg>...	Delete a default kernel argument from the installed system.
-n, --copy-network	Copy the network configuration from the install environment. <div style="display: flex; align-items: flex-start;"> <div style="width: 40px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>The --copy-network option only copies networking configuration found under /etc/NetworkManager/system-connections. In particular, it does not copy the system hostname.</p> </div> </div>
--network-dir <path>	For use with -n . Default is /etc/NetworkManager/system-connections/ .
--save-partlabel <lx>..	Save partitions with this label glob.

--save-partindex <id>...	Save partitions with this number or range.
--insecure	Skip RHCOS image signature verification.
--insecure-ignition	Allow Ignition URL without HTTPS or hash.
--architecture <name>	Target CPU architecture. Valid values are x86_64 and aarch64 .
--preserve-on-error	Do not clear partition table on error.
-h, --help	Print help information.
coreos-installer install subcommand argument	
<i>Argument</i>	<i>Description</i>
<device>	The destination device.
coreos-installer ISO subcommands	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer iso customize <options> <ISO_image>	Customize a RHCOS live ISO image.
coreos-installer iso reset <options> <ISO_image>	Restore a RHCOS live ISO image to default settings.
coreos-installer iso ignition remove <options> <ISO_image>	Remove the embedded Ignition config from an ISO image.
coreos-installer ISO customize subcommand options	
<i>Option</i>	<i>Description</i>
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--dest-karg-append <arg>	Add a kernel argument to each boot of the destination system.

--dest-karg-delete <arg>	Delete a kernel argument from each boot of the destination system.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
--post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
--live-karg-append <arg>	Add a kernel argument to each boot of the live environment.
--live-karg-delete <arg>	Delete a kernel argument from each boot of the live environment.
--live-karg-replace <k=o=n>	Replace a kernel argument in each boot of the live environment, in the form key=old=new .
-f, --force	Overwrite an existing Ignition config.
-o, --output <path>	Write the ISO to a new output file.
-h, --help	Print help information.
coreos-installer PXE subcommands	
<i>Subcommand</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
coreos-installer pxe customize <options> <path>	Customize a RHCOS live PXE boot config.
coreos-installer pxe ignition wrap <options>	Wrap an Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <image_name>	Show the wrapped Ignition config in an image.

coreos-installer PXE customize subcommand options	
<i>Option</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
-o, --output <path>	Write the initramfs to a new output file. <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>This option is required for PXE environments.</p> </div> </div>
-h, --help	Print help information.

12.2.11.3.9.3. coreos.inst boot options for ISO or PXE installations

You can automatically invoke **coreos-installer** options at boot time by passing **coreos.inst** boot arguments to the RHCOS live installer. These are provided in addition to the standard boot arguments.

- For ISO installations, the **coreos.inst** options can be added by interrupting the automatic boot at the bootloader menu. You can interrupt the automatic boot by pressing **TAB** while the **RHEL CoreOS (Live)** menu option is highlighted.

- For PXE or IPXE installations, the **coreos.inst** options must be added to the **APPEND** line before the RHCOS live installer is booted.

The following table shows the RHCOS live installer **coreos.inst** boot options for ISO and PXE installations.

Table 12.13. coreos.inst boot options

Argument	Description
coreos.inst.install_dev	Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda , although sda is allowed.
coreos.inst.ignition_url	Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded. Only HTTP and HTTPS protocols are supported.
coreos.inst.save_partlabel	Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.
coreos.inst.save_partindex	Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.
coreos.inst.insecure	Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.
coreos.inst.image_url	Optional: Download and install the specified RHCOS image. <ul style="list-style-type: none"> • This argument should not be used in production environments and is intended for debugging purposes only. • While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install. • If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform. • Only HTTP and HTTPS protocols are supported.

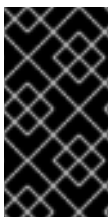
Argument	Description
coreos.inst.skip_reboot	Optional: The system will not reboot after installing. After the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.
coreos.inst.platform_id	Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal . This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: coreos.inst.platform_id=vmware.
ignition.config.url	Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from coreos.inst.ignition_url , which is the Ignition config for the installed system.

12.2.11.4. Enabling multipathing with kernel arguments on RHCOS

RHCOS supports multipathing on the primary disk, allowing stronger resilience to hardware failure to achieve higher host availability.

You can enable multipathing at installation time for nodes that were provisioned in OpenShift Container Platform 4.8 or later. While postinstallation support is available by activating multipathing via the machine config, enabling multipathing during installation is recommended.

In setups where any I/O to non-optimized paths results in I/O system errors, you must enable multipathing at installation time.



IMPORTANT

On IBM Z and IBM® LinuxONE, you can enable multipathing only if you configured your cluster for it during installation. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process" in *Installing a cluster with z/VM on IBM Z and IBM® LinuxONE*.

The following procedure enables multipath at installation time and appends kernel arguments to the **coreos-installer install** command so that the installed system itself will use multipath beginning from the first boot.



NOTE

OpenShift Container Platform does not support enabling multipathing as a day-2 activity on nodes that have been upgraded from 4.6 or earlier.

Procedure

1. To enable multipath and start the **multipathd** daemon, run the following command on the installation host:

```
$ mpathconf --enable && systemctl start multipathd.service
```

- Optional: If booting the PXE or ISO, you can instead enable multipath by adding **rd.multipath=default** from the kernel command line.

2. Append the kernel arguments by invoking the **coreos-installer** program:

- If there is only one multipath device connected to the machine, it should be available at path **/dev/mapper/mpatha**. For example:

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

- 1 Indicates the path of the single multipathed device.

- If there are multiple multipath devices connected to the machine, or to be more explicit, instead of using **/dev/mapper/mpatha**, it is recommended to use the World Wide Name (WWN) symlink available in **/dev/disk/by-id**. For example:

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

- 1 Indicates the WWN ID of the target multipathed device. For example, **0xx194e957fcedb4841**.

This symlink can also be used as the **coreos.inst.install_dev** kernel argument when using special **coreos.inst.*** arguments to direct the live installer. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process".

3. Check that the kernel arguments worked by going to one of the worker nodes and listing the kernel command line arguments (in **/proc/cmdline** on the host):

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

Example output

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

You should see the added kernel arguments.

Additional resources

- See [Installing RHCOS and starting the OpenShift Container Platform bootstrap process](#) for more information on using special **coreos.inst.*** arguments to direct the live installer.

12.2.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1  
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

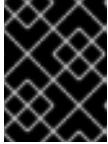
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...  
INFO API v1.25.0 up  
INFO Waiting up to 30m0s for bootstrapping to complete...  
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

Additional resources

- See [Monitoring installation progress](#) for more information about monitoring the installation logs and retrieving diagnostic data if installation issues arise.

12.2.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

12.2.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

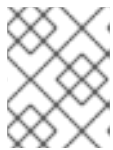
1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

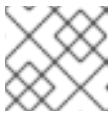
- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

12.2.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

- Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m

console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

Additional resources

- See [Gathering logs from a failed installation](#) for details about gathering data in the event of a failed OpenShift Container Platform installation.
- See [Troubleshooting Operator issues](#) for steps to check Operator pod health across the cluster and gather Operator logs for diagnosis.

12.2.15.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

12.2.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

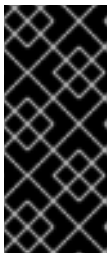
Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

12.2.15.2.1. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

12.2.15.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

12.2.15.2.3. Configuring block registry storage for bare metal

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes, or block persistent volumes, are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

If you choose to use a block storage volume with the image registry, you must use a filesystem persistent volume claim (PVC).

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only one (**1**) replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
    - ReadWriteOnce 3
```



```
resources:
  requests:
    storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

12.2.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

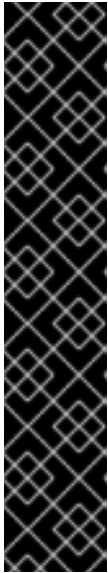
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8  1/1
Running   0    5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

12.2.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

12.2.18. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .

12.3. INSTALLING A USER-PROVISIONED BARE METAL CLUSTER WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform 4.12, you can install a cluster on bare metal infrastructure that you provision with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

When you customize OpenShift Container Platform networking, you must set most of the network configuration parameters during installation. You can modify only **kubeProxy** network configuration parameters in a running cluster.

12.3.1. Prerequisites

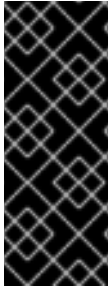
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.

12.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

Additional resources

- See [Installing a user-provisioned bare metal cluster on a restricted network](#) for more information about performing a restricted network installation on bare metal infrastructure that you provision.

12.3.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

12.3.3.1. Required machines for cluster installation

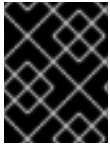
The smallest OpenShift Container Platform clusters require the following hosts:

Table 12.14. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.

**NOTE**

As an exception, you can run zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production. Running one compute machine is not supported.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

12.3.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 12.15. Minimum resource requirements

Machine	Operating System	CPU [1]	RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

12.3.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- See [Configuring a three-node cluster](#) for details about deploying three-node clusters in bare metal environments.
- See [Approving the certificate signing requests for your machines](#) for more information about approving cluster certificate signing requests after installation.

12.3.3.4. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

12.3.3.4.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

12.3.3.4.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 12.16. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 12.17. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 12.18. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

12.3.3.5. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 12.19. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
Routes	*.apps.<cluster_name>.<base_domain>.	 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>
		<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>

Component	Record	Description
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

12.3.3.5.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 12.4. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
```

```

;
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

-

Example 12.5. Sample DNS zone database for reverse records

```

$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF

```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

- [Validating DNS resolution for user-provisioned infrastructure](#)

12.3.3.6. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 12.20. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 12.21. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

12.3.3.6.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 12.6. Sample API and application Ingress load balancer configuration

```

global
log      127.0.0.1 local2
pidfile  /var/run/haproxy.pid
maxconn  4000
daemon

defaults
mode            http
log             global
option          dontlognull
option http-server-close
option          redispatch
retries         3
timeout http-request 10s
timeout queue   1m
timeout connect 10s
timeout client  1m
timeout server  1m
timeout http-keep-alive 10s
timeout check   10s
maxconn         3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```


- 1 Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2 4 The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3 Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

12.3.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.

- b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.

**NOTE**

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.

- b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

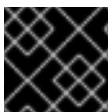
Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

Additional resources

- [Requirements for a cluster with user-provisioned infrastructure](#)
- [Installing RHCOS and starting the OpenShift Container Platform bootstrap process](#)
- [Setting the cluster node hostnames through DHCP](#)
- [Advanced RHCOS installation configuration](#)
- [Networking requirements for user-provisioned infrastructure](#)
- [User-provisioned DNS requirements](#)
- [Validating DNS resolution for user-provisioned infrastructure](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.3.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1 Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1** Provides the record name for the Kubernetes internal API.
- 2** Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

Additional resources

- [User-provisioned DNS requirements](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.3.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1** Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

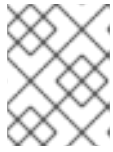
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

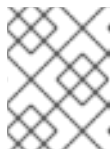
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

Additional resources

- [Verifying node health](#)

12.3.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

12.3.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.

3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

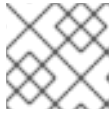
Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

12.3.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

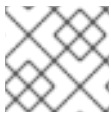
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

12.3.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

12.3.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 12.22. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

12.3.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.

- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.

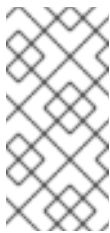
```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 12.23. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23 - cidr: fd01::/48 hostPrefix: 64</pre>


Parameter	Description	Values
networking.clusterNetwork.cidr	<p>Required if you use networking.clusterNetwork. An IP address block.</p> <p>If you use the OpenShift SDN network plugin, specify an IPv4 network. If you use the OVN-Kubernetes network plugin, you can specify IPv4 and IPv6 networks.</p>	<p>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32. The prefix length for an IPv6 block is between 0 and 128. For example, 10.128.0.0/14 or fd01::/48.</p>
networking.clusterNetwork.hostPrefix	<p>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.</p>	<p>A subnet prefix.</p> <p>For an IPv4 network the default value is 23. For an IPv6 network the default value is 64. The default value is also the minimum value for IPv6.</p>
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p> <p>If you use the OVN-Kubernetes network plugin, you can specify an IP address block for both of the IPv4 and IPv6 address families.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16 - fd02::/112</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16 or fd00::/48.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


12.3.9.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 12.24. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String

Parameter	Description	Values
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.

Parameter	Description	Values
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 50px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...

12.3.9.2. Sample install-config.yaml file for bare metal

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
metadata:
  name: test 8

```

```

networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14 9
      hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
    - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.

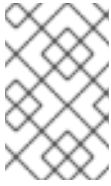


NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.

- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for your platform.



IMPORTANT

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

Additional resources

- See [Load balancing requirements for user-provisioned infrastructure](#) for more information on the API and application ingress load balancing requirements.

12.3.10. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.

**NOTE**

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

**IMPORTANT**

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

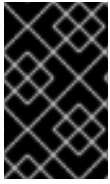
Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

12.3.11. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

- Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

- Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```



```

defaultNetwork:
  ovnKubernetesConfig:
    ipsecConfig: {}

```

- Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

12.3.12. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

12.3.12.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 12.25. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre> spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23 </pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 12.26. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 12.27. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 12.28. ovnKubernetesConfig object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is 2⁽²³⁻¹⁴⁾=512.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. This field cannot be changed after installation.	The default value is fd98::/48 .

Table 12.29. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 12.30. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 12.31. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

12.3.13. Creating the Ignition config files

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

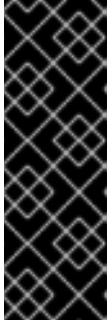
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

- Obtain the Ignition config files:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.



IMPORTANT

If you created an **install-config.yaml** file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

The following files are generated in the directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

12.3.14. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.



NOTE

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Only RHEL 8 compute machines are supported.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments:** You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the **APPEND** parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special **coreos.inst.*** arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.
- **Ignition configs:** OpenShift Container Platform Ignition config files (***.ign**) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the **coreos-installer** to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.
- **coreos-installer:** You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the **coreos-installer** command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.



NOTE

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

12.3.14.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

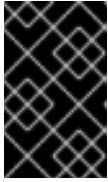
Procedure

1. Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

2. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

3. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total % Received % Xferd Average Speed Time Time Time Current
           Dload Upload Total Spent Left Speed
0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

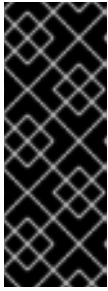
Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

4. Although it is possible to obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS images are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep '\.iso[^\.]'
```

Example output

```
"location": "<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:
 - Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.



NOTE

It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

- 1** You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.
- 2** The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.



NOTE

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the **/dev/sda** device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
```

```
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

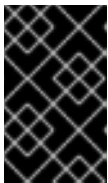
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.3.14.2. Installing RHCOS by using PXE or iPXE booting

You can use PXE or iPXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE or iPXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left  Speed
  0  0  0  0  0  0  0  0  0  --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

3. Although it is possible to obtain the RHCOS **kernel**, **initramfs** and **rootfs** files that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS files are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep -Eo "https.*(kernel-|initramfs-|rootfs-)\w+(\.img)?"
```

Example output

```
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
```

```

rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"

```



IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** rhcos-<version>-live-kernel-<architecture>
- **initramfs:** rhcos-<version>-live-initramfs.<architecture>.img
- **rootfs:** rhcos-<version>-live-rootfs.<architecture>.img

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

6. Configure PXE or iPXE installation for the RHCOS images and begin the installation. Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE (**x86_64**):

```

DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3

```

- 1 1 Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

- For iPXE (**x86_64** + **aarch64**):

```

kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot

```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **kernel** parameter value is the location of the **kernel** file, the **initrd=main** argument is needed for booting on UEFI systems, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.

- 3 Specify the location of the **initramfs** file that you uploaded to your HTTP server.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **kernel** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.



NOTE

To network boot the CoreOS **kernel** on **aarch64** architecture, you need to use a version of iPXE build with the **IMAGE_GZIP** option enabled. See [IMAGE_GZIP option in iPXE](#).

- For PXE (with UEFI and Grub as second stage) on **aarch64**:

```
menuentry 'Install CoreOS' {
  linux rhcos-<version>-live-kernel-<architecture>
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
  <architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
  initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}
```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP/TFTP server. The **kernel** parameter value is the location of the **kernel** file on your TFTP server. The **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file on your HTTP Server.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your TFTP server.

7. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

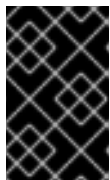
8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.

9. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. Continue to create the machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.3.14.3. Advanced RHCOS installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
- Customizing a live ISO or PXE boot image

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

12.3.14.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

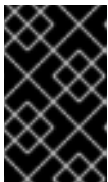
- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

Procedure

1. Boot the ISO installer.
2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as **nmcli** or **nmtui**.
3. Run the **coreos-installer** command to install the system, adding the **--copy-network** option to copy networking configuration. For example:

```
$ sudo coreos-installer install --copy-network \
  --ignition-url=http://host/worker.ign /dev/sda
```



IMPORTANT

The **--copy-network** option only copies networking configuration found under **/etc/NetworkManager/system-connections**. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

Additional resources

- See [Getting started with nmcli](#) and [Getting started with nmtui](#) in the RHEL 8 documentation for more information about the **nmcli** and **nmtui** tools.

12.3.14.3.2. Disk partitioning

Disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless you override the default partitioning configuration. During the RHCOS installation, the size of the root file system is increased to use any remaining available space on the target device.



IMPORTANT

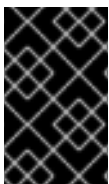
The use of a custom partition scheme on your node might result in OpenShift Container Platform not monitoring or alerting on some node partitions. If you override the default partitioning, see [Understanding OpenShift File System Monitoring \(eviction conditions\)](#) for more information about how OpenShift Container Platform monitors your host file systems.

OpenShift Container Platform monitors the following two filesystem identifiers:

- **nodefs**, which is the filesystem that contains **/var/lib/kubelet**
- **imagefs**, which is the filesystem that contains **/var/lib/containers**

For the default partition scheme, **nodefs** and **imagefs** monitor the same root filesystem, **/**.

To override the default partitioning when installing RHCOS on an OpenShift Container Platform cluster node, you must create separate partitions.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.

Consider a situation where you want to add a separate storage partition for your containers and container images. For example, by mounting **/var/lib/containers** in a separate partition, the kubelet separately monitors **/var/lib/containers** as the **imagefs** directory and the root file system as the **nodefs** directory.



IMPORTANT

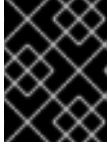
If you have resized your disk size to host a larger file system, consider creating a separate **/var/lib/containers** partition. Consider resizing a disk that has an **xfs** format to reduce CPU time issues caused by a high number of allocation groups.

12.3.14.3.2.1. Creating a separate **/var** partition

In general, you should use the default disk partitioning that is created during the RHCOS installation. However, there are cases where you might want to create a separate partition for a directory that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** directory or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate `/var` partition.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

The use of a separate partition for the `/var` directory or a subdirectory of `/var` also prevents data growth in the partitioned directory from filling up the root file system.

The following procedure sets up a separate `/var` partition by adding a machine config manifest that is wrapped into the Ignition config file for a node type during the preparation phase of an installation.

Procedure

1. On your installation host, change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ openshift-install create manifests --dir <installation_directory>
```

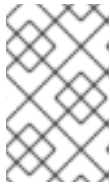
2. Create a Butane config that configures the additional partition. For example, name the file `$HOME/clusterconfig/98-var-partition.bu`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum offset value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no offset value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future

reinstalls of RHCOS might overwrite the beginning of the data partition.

- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for compute nodes, if the different instance types do not have the same device name.

3. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. Create the Ignition config files:

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

The files in the **<installation_directory>/manifest** and **<installation_directory>/openshift** directories are wrapped into the Ignition config files, including the file that contains the **98-var-partition** custom **MachineConfig** object.

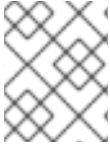
Next steps

- You can apply the custom disk partitioning by referencing the Ignition config files during the RHCOS installations.

12.3.14.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the **coreos-installer** command that cause the installer to maintain one or more existing partitions. For a PXE installation, you can add **coreos.inst.*** options to the **APPEND** parameter to preserve partitions.

Saved partitions might be data partitions from an existing OpenShift Container Platform system. You can identify the disk partitions you want to keep either by partition label or by number.



NOTE

If you save existing partitions, and those partitions do not leave enough space for RHCOS, the installation will fail without damaging the saved partitions.

Retaining existing partitions during an ISO installation

This example preserves any partition in which the partition label begins with **data** (**data***):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the **coreos-installer** in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
  --save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, **coreos-installer** recreates the partition immediately.

Retaining existing partitions during a PXE installation

This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

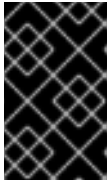
This **APPEND** option preserves partition 6:

```
coreos.inst.save_partindex=6
```

12.3.14.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config:** Every manual RHCOS installation needs to pass one of the Ignition config files generated by **openshift-installer**, such as **bootstrap.ign**, **master.ign** and **worker.ign**, to carry out the installation.



IMPORTANT

It is not recommended to modify these Ignition config files directly. You can update the manifest files that are wrapped into the Ignition config files, as outlined in examples in the preceding sections.

For PXE installations, you pass the Ignition configs on the **APPEND** line using the **coreos.inst.ignition_url=** option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the **coreos-installer** command line with the **--ignition-url=** option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config:** This type can be created by using the **coreos-installer customize** subcommand and its various options. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be done once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

For PXE or ISO boots, you can create the Ignition config and **APPEND** the **ignition.config.url=** option to identify the location of the Ignition config. You also need to append **ignition.firstboot** or the **ignition.platform.id=metal** option will be ignored.

12.3.14.3.4. Default console configuration

Red Hat Enterprise Linux CoreOS (RHCOS) nodes installed from an OpenShift Container Platform 4.12 boot image use a default console that is meant to accommodate most virtualized and bare metal setups. Different cloud and virtualization platforms may use different default settings depending on the chosen architecture. Bare metal installations use the kernel default settings which typically means the graphical console is the primary console and the serial console is disabled.

The default consoles may not match your specific hardware configuration or you might have specific needs that require you to adjust the default console. For example:

- You want to access the emergency shell on the console for debugging purposes.
- Your cloud platform does not provide interactive access to the graphical console, but provides a serial console.
- You want to enable multiple consoles.

Console configuration is inherited from the boot image. This means that new nodes in existing clusters are unaffected by changes to the default console.

You can configure the console for bare metal installations in the following ways:

- Using **coreos-installer** manually on the command line.
- Using the **coreos-installer iso customize** or **coreos-installer pxe customize** subcommands with the **--dest-console** option to create a custom image that automates the process.



NOTE

For advanced customization, perform console configuration using the **coreos-installer iso** or **coreos-installer pxe** subcommands, and not kernel arguments.

12.3.14.3.5. Enabling the serial console for PXE and ISO installations

By default, the Red Hat Enterprise Linux CoreOS (RHCOS) serial console is disabled and all output is written to the graphical console. You can enable the serial console for an ISO installation and reconfigure the bootloader so that output is sent to both the serial console and the graphical console.

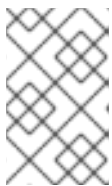
Procedure

1. Boot the ISO installer.
2. Run the **coreos-installer** command to install the system, adding the **--console** option once to specify the graphical console, and a second time to specify the serial console:

```
$ coreos-installer install \
  --console=tty0 \ 1
  --console=ttyS0,<options> \ 2
  --ignition-url=http://host/worker.ign /dev/sda
```

- 1 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 2 The desired primary console. In this case the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **11520n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see [Linux kernel serial console](#) documentation.

3. Reboot into the installed system.



NOTE

A similar outcome can be obtained by using the **coreos-installer install --append-karg** option, and specifying the console with **console=**. However, this will only set the console for the kernel and not the bootloader.

To configure a PXE installation, make sure the **coreos.inst.install_dev** kernel command line option is omitted, and use the shell prompt to run **coreos-installer** manually using the above ISO installation procedure.

12.3.14.3.6. Customizing a live RHCOS ISO or PXE install

You can use the live ISO image or PXE environment to install RHCOS by injecting an Ignition config file directly into the image. This creates a customized image that you can use to provision your system.

For an ISO image, the mechanism to do this is the **coreos-installer iso customize** subcommand, which modifies the **.iso** file with your configuration. Similarly, the mechanism for a PXE environment is the **coreos-installer pxe customize** subcommand, which creates a new **initramfs** file that includes your customizations.

The **customize** subcommand is a general purpose tool that can embed other types of customizations as well. The following tasks are examples of some of the more common customizations:

- Inject custom CA certificates for when corporate security policy requires their use.
- Configure network settings without the need for kernel arguments.
- Embed arbitrary preinstall and post-install scripts or binaries.

12.3.14.3.7. Customizing a live RHCOS ISO image

You can customize a live RHCOS ISO image directly with the **coreos-installer iso customize** subcommand. When you boot the ISO image, the customizations are applied automatically.

You can use this feature to configure the ISO image to automatically install RHCOS.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to inject the Ignition config directly into the ISO image:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda 2
```

- 1 The Ignition config file that is generated from the **openshift-installer** installation program.
- 2 When you specify this option, the ISO image automatically runs an installation. Otherwise, the image remains configured for installation, but does not install automatically unless you specify the **coreos.inst.install_dev** kernel argument.

3. Optional: To remove the ISO image customizations and return the image to its pristine state, run:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now re-customize the live ISO image or use it in its pristine state.

Applying your customizations affects every subsequent boot of RHCOS.

12.3.14.3.7.1. Modifying a live install ISO image to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

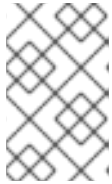
Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image to enable the serial console to receive output:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/sda 4
```

- 1 The location of the Ignition config to install.

- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. In this case, **/dev/sda**. If you omit this option, the ISO image automatically runs the installation program which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.



NOTE

The **--dest-console** option affects the installed system and not the live ISO system. To modify the console for a live ISO system, use the **--live-karg-append** option and specify the console with **console=**.

Your customizations are applied and affect every subsequent boot of the ISO image.

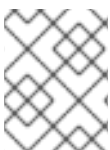
3. Optional: To remove the ISO image customizations and return the image to its original state, run the following command:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now recustomize the live ISO image or use it in its original state.

12.3.14.3.7.2. Modifying a live install ISO image to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



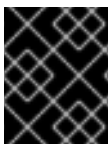
NOTE

Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image for use with a custom CA:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```



IMPORTANT

The **coreos.inst.ignition_url** kernel parameter does not work with the **--ignition-ca** flag. You must use the **--dest-ignition** flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.3.14.3.7.3. Modifying a live install ISO image with customized network settings

You can embed a NetworkManager keyfile into the live ISO image and pass it through to the installed system with the **--network-keyfile** flag of the **customize** subcommand.



WARNING

When creating a connection profile, you must use a **.nmconnection** filename extension in the filename of the connection profile. If you do not use a **.nmconnection** filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Create a connection profile for a bonded interface. For example, create the **bond0.nmconnection** file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
```

```

interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=

```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```

[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=

```

5. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image with your configured networking:

```

$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection

```

Network settings are applied to the live system and are carried over to the destination system.

12.3.14.3.8. Customizing a live RHCOS PXE environment

You can customize a live RHCOS PXE environment directly with the **coreos-installer pxe customize** subcommand. When you boot the PXE environment, the customizations are applied automatically.

You can use this feature to configure the PXE environment to automatically install RHCOS.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new **initramfs** file that contains the customizations from your Ignition config:

```

$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3

```

- 1 The Ignition config file that is generated from **openshift-installer**.
- 2 When you specify this option, the PXE environment automatically runs an install. Otherwise, the image remains configured for installing, but does not do so automatically unless you specify the **coreos.inst.install_dev** kernel argument.
- 3 Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Applying your customizations affects every subsequent boot of RHCOS.

12.3.14.3.8.1. Modifying a live install PXE environment to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new customized **initramfs** file that enables the serial console to receive output:

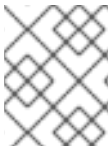
```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--dest-ignition <path> \ 1
--dest-console tty0 \ 2
--dest-console ttyS0,<options> \ 3
--dest-device /dev/sda \ 4
-o rhcos-<version>-custom-initramfs.x86_64.img 5
```

- 1 The location of the Ignition config to install.
- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. If you omit this option, the PXE environment automatically runs the installer which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.
- 5 Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Your customizations are applied and affect every subsequent boot of the PXE environment.

12.3.14.3.8.2. Modifying a live install PXE environment to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



NOTE

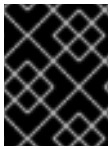
Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file for use with a custom CA:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --ignition-ca cert.pem \
  -o rhcos-<version>-custom-initramfs.x86_64.img
```

3. Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.



IMPORTANT

The **coreos.inst.ignition_url** kernel parameter does not work with the **--ignition-ca** flag. You must use the **--dest-ignition** flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.3.14.3.8.3. Modifying a live install PXE environment with customized network settings

You can embed a NetworkManager keyfile into the live PXE environment and pass it through to the installed system with the **--network-keyfile** flag of the **customize** subcommand.



WARNING

When creating a connection profile, you must use a **.nmconnection** filename extension in the filename of the connection profile. If you do not use a **.nmconnection** filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Create a connection profile for a bonded interface. For example, create the **bond0.nmconnection** file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file that contains your configured networking:


```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--network-keyfile bond0.nmconnection \
--network-keyfile bond0-proxy-em1.nmconnection \
--network-keyfile bond0-proxy-em2.nmconnection \
-o rhcos-<version>-custom-initramfs.x86_64.img
```

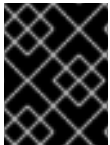
- Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present. Network settings are applied to the live system and are carried over to the destination system.

12.3.14.3.9. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

12.3.14.3.9.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**

- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option. Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

12.3.14.3.9.2. coreos-installer options for ISO and PXE installations

You can install RHCOS by running **coreos-installer install <options> <device>** at the command prompt, after booting into the RHCOS live environment from an ISO image.

The following table shows the subcommands, options, and arguments you can pass to the **coreos-installer** command.


Table 12.32. coreos-installer subcommands, command-line options, and arguments

coreos-installer install subcommand

<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer install <options> <device>	Embed an Ignition config in an ISO image.
coreos-installer install subcommand options	
<i>Option</i>	<i>Description</i>
-u, --image-url <url>	Specify the image URL manually.
-f, --image-file <path>	Specify a local image file manually. Used for debugging.
-i, --ignition-file <path>	Embed an Ignition config from a file.
-l, --ignition-url <URL>	Embed an Ignition config from a URL.
--ignition-hash <digest>	Digest type-value of the Ignition config.
-p, --platform <name>	Override the Ignition platform ID for the installed system.
--console <spec>	Set the kernel and bootloader console for the installed system. For more information about the format of <spec> , see the Linux kernel serial console documentation.
--append-karg <arg>...	Append a default kernel argument to the installed system.
--delete-karg <arg>...	Delete a default kernel argument from the installed system.
-n, --copy-network	Copy the network configuration from the install environment. <div data-bbox="815 1597 922 1854" style="background-color: black; width: 67px; height: 115px; margin: 10px 0;"></div> <div data-bbox="1002 1603 1190 1637" style="font-weight: bold; margin: 10px 0;">IMPORTANT</div> <div data-bbox="1002 1671 1406 1854" style="margin: 10px 0;">The --copy-network option only copies networking configuration found under /etc/NetworkManager/system-connections. In particular, it does not copy the system hostname.</div>
--network-dir <path>	For use with -n . Default is /etc/NetworkManager/system-connections/ .
--save-partlabel <lx>..	Save partitions with this label glob.

--save-partindex <id>...	Save partitions with this number or range.
--insecure	Skip RHCOS image signature verification.
--insecure-ignition	Allow Ignition URL without HTTPS or hash.
--architecture <name>	Target CPU architecture. Valid values are x86_64 and aarch64 .
--preserve-on-error	Do not clear partition table on error.
-h, --help	Print help information.
coreos-installer install subcommand argument	
<i>Argument</i>	<i>Description</i>
<device>	The destination device.
coreos-installer ISO subcommands	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer iso customize <options> <ISO_image>	Customize a RHCOS live ISO image.
coreos-installer iso reset <options> <ISO_image>	Restore a RHCOS live ISO image to default settings.
coreos-installer iso ignition remove <options> <ISO_image>	Remove the embedded Ignition config from an ISO image.
coreos-installer ISO customize subcommand options	
<i>Option</i>	<i>Description</i>
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--dest-karg-append <arg>	Add a kernel argument to each boot of the destination system.

--dest-karg-delete <arg>	Delete a kernel argument from each boot of the destination system.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
--post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
--live-karg-append <arg>	Add a kernel argument to each boot of the live environment.
--live-karg-delete <arg>	Delete a kernel argument from each boot of the live environment.
--live-karg-replace <k=o=n>	Replace a kernel argument in each boot of the live environment, in the form key=old=new .
-f, --force	Overwrite an existing Ignition config.
-o, --output <path>	Write the ISO to a new output file.
-h, --help	Print help information.
coreos-installer PXE subcommands	
<i>Subcommand</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
coreos-installer pxe customize <options> <path>	Customize a RHCOS live PXE boot config.
coreos-installer pxe ignition wrap <options>	Wrap an Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <image_name>	Show the wrapped Ignition config in an image.

coreos-installer PXE customize subcommand options	
<i>Option</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
-o, --output <path>	Write the initramfs to a new output file.
	 <p>NOTE</p> <p>This option is required for PXE environments.</p>
-h, --help	Print help information.

12.3.14.3.9.3. coreos.inst boot options for ISO or PXE installations

You can automatically invoke **coreos-installer** options at boot time by passing **coreos.inst** boot arguments to the RHCOS live installer. These are provided in addition to the standard boot arguments.

- For ISO installations, the **coreos.inst** options can be added by interrupting the automatic boot at the bootloader menu. You can interrupt the automatic boot by pressing **TAB** while the **RHEL CoreOS (Live)** menu option is highlighted.

- For PXE or IPXE installations, the **coreos.inst** options must be added to the **APPEND** line before the RHCOS live installer is booted.

The following table shows the RHCOS live installer **coreos.inst** boot options for ISO and PXE installations.

Table 12.33. coreos.inst boot options

Argument	Description
coreos.inst.install_dev	Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda , although sda is allowed.
coreos.inst.ignition_url	Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded. Only HTTP and HTTPS protocols are supported.
coreos.inst.save_partlabel	Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.
coreos.inst.save_partindex	Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.
coreos.inst.insecure	Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.
coreos.inst.image_url	Optional: Download and install the specified RHCOS image. <ul style="list-style-type: none"> • This argument should not be used in production environments and is intended for debugging purposes only. • While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install. • If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform. • Only HTTP and HTTPS protocols are supported.

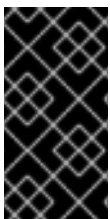
Argument	Description
coreos.inst.skip_reboot	Optional: The system will not reboot after installing. After the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.
coreos.inst.platform_id	Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal . This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: coreos.inst.platform_id=vmware.
ignition.config.url	Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from coreos.inst.ignition_url , which is the Ignition config for the installed system.

12.3.14.4. Enabling multipathing with kernel arguments on RHCOS

RHCOS supports multipathing on the primary disk, allowing stronger resilience to hardware failure to achieve higher host availability.

You can enable multipathing at installation time for nodes that were provisioned in OpenShift Container Platform 4.8 or later. While postinstallation support is available by activating multipathing via the machine config, enabling multipathing during installation is recommended.

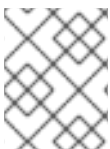
In setups where any I/O to non-optimized paths results in I/O system errors, you must enable multipathing at installation time.



IMPORTANT

On IBM Z and IBM® LinuxONE, you can enable multipathing only if you configured your cluster for it during installation. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process" in *Installing a cluster with z/VM on IBM Z and IBM® LinuxONE*.

The following procedure enables multipath at installation time and appends kernel arguments to the **coreos-installer install** command so that the installed system itself will use multipath beginning from the first boot.



NOTE

OpenShift Container Platform does not support enabling multipathing as a day-2 activity on nodes that have been upgraded from 4.6 or earlier.

Procedure

1. To enable multipath and start the **multipathd** daemon, run the following command on the installation host:

```
$ mpathconf --enable && systemctl start multipathd.service
```

- Optional: If booting the PXE or ISO, you can instead enable multipath by adding **rd.multipath=default** from the kernel command line.

2. Append the kernel arguments by invoking the **coreos-installer** program:

- If there is only one multipath device connected to the machine, it should be available at path **/dev/mapper/mpatha**. For example:

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

- 1 Indicates the path of the single multipathed device.

- If there are multiple multipath devices connected to the machine, or to be more explicit, instead of using **/dev/mapper/mpatha**, it is recommended to use the World Wide Name (WWN) symlink available in **/dev/disk/by-id**. For example:

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

- 1 Indicates the WWN ID of the target multipathed device. For example, **0xx194e957fcedb4841**.

This symlink can also be used as the **coreos.inst.install_dev** kernel argument when using special **coreos.inst.*** arguments to direct the live installer. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process".

3. Check that the kernel arguments worked by going to one of the worker nodes and listing the kernel command line arguments (in **/proc/cmdline** on the host):

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

Example output

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

You should see the added kernel arguments.

12.3.15. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1  
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...  
INFO API v1.25.0 up  
INFO Waiting up to 30m0s for bootstrapping to complete...  
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

Additional resources

- See [Monitoring installation progress](#) for more information about monitoring the installation logs and retrieving diagnostic data if installation issues arise.

12.3.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

12.3.17. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```

NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 63m  v1.25.0
master-1  Ready   master 63m  v1.25.0
master-2  Ready   master 64m  v1.25.0

```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```

NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...

```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

12.3.18. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

- Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m

console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

Additional resources

- See [Gathering logs from a failed installation](#) for details about gathering data in the event of a failed OpenShift Container Platform installation.
- See [Troubleshooting Operator issues](#) for steps to check Operator pod health across the cluster and gather Operator logs for diagnosis.

12.3.18.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

12.3.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

12.3.18.3. Configuring block registry storage for bare metal

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes, or block persistent volumes, are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

If you choose to use a block storage volume with the image registry, you must use a filesystem persistent volume claim (PVC).

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only one (**1**) replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
    - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

12.3.19. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m

kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8    1/1
Running   0    5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

12.3.20. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

12.3.21. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .

12.4. INSTALLING A USER-PROVISIONED BARE METAL CLUSTER ON A RESTRICTED NETWORK

In OpenShift Container Platform 4.12, you can install a cluster on bare metal infrastructure that you provision in a restricted network.



IMPORTANT

While you might be able to follow this procedure to deploy a cluster on virtualized or cloud environments, you must be aware of additional considerations for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you attempt to install an OpenShift Container Platform cluster in such an environment.

12.4.1. Prerequisites

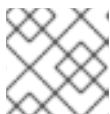
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

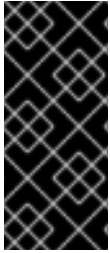
12.4.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on

the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

12.4.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

12.4.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

12.4.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

12.4.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 12.34. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



NOTE

As an exception, you can run zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production. Running one compute machine is not supported.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

12.4.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 12.35. Minimum resource requirements

Machine	Operating System	CPU [1]	RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

12.4.4.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- See [Configuring a three-node cluster](#) for details about deploying three-node clusters in bare metal environments.
- See [Approving the certificate signing requests for your machines](#) for more information about approving cluster certificate signing requests after installation.

12.4.4.4. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

12.4.4.4.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

12.4.4.4.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

Table 12.36. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 12.37. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 12.38. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP)

server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

12.4.4.5. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 12.39. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
	api-int.<cluster_name>.<base_domain>.	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

12.4.4.5.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 12.7. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines

by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 12.8. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.

4 5 6 Provides reverse DNS resolution for the control plane machines.

7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

Additional resources

- [Validating DNS resolution for user-provisioned infrastructure](#)

12.4.4.6. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

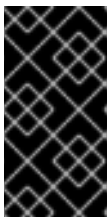


NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer:** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 12.40. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 12.41. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

12.4.4.6.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 12.9. Sample API and application Ingress load balancer configuration

```
global
  log 127.0.0.1 local2
  pidfile /var/run/haproxy.pid
  maxconn 4000
  daemon
defaults
  mode http
  log global
  option dontlognull
  option http-server-close
  option redispatch
  retries 3
  timeout http-request 10s
  timeout queue 1m
  timeout connect 10s
  timeout client 1m
```

```

timeout server      1m
timeout http-keep-alive 10s
timeout check       10s
maxconn             3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

12.4.5. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

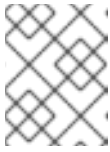
Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

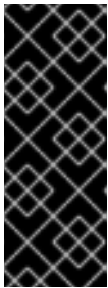
If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.

**NOTE**

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

Additional resources

- [Requirements for a cluster with user-provisioned infrastructure](#)
- [Installing RHCOS and starting the OpenShift Container Platform bootstrap process](#)
- [Setting the cluster node hostnames through DHCP](#)
- [Advanced RHCOS installation configuration](#)
- [Networking requirements for user-provisioned infrastructure](#)
- [User-provisioned DNS requirements](#)
- [Validating DNS resolution for user-provisioned infrastructure](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.4.6. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.

- a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1** Provides the record name for the Kubernetes internal API.
- 2** Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

Additional resources

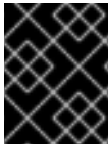
- [User-provisioned DNS requirements](#)
- [Load balancing requirements for user-provisioned infrastructure](#)

12.4.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

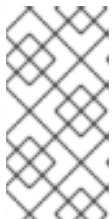
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

■

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

Additional resources

- [Verifying node health](#)

12.4.8. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

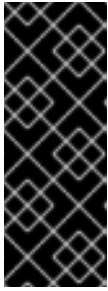
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain the **imageContentSources** section from the output of the command to mirror the repository.
- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

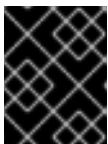


NOTE

You must name this configuration file **install-config.yaml**.

- Unless you use a registry that RHCOS trusts by default, such as **docker.io**, you must provide the contents of the certificate for your mirror repository in the **additionalTrustBundle** section. In most cases, you must provide the certificate for your mirror.
- You must include the **imageContentSources** section from the output of the command to mirror the repository.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

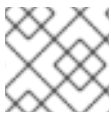


IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

12.4.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

12.4.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 12.42. Required parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

12.4.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.


```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 12.43. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23 - cidr: fd01::/48 hostPrefix: 64</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. If you use the OpenShift SDN network plugin, specify an IPv4 network. If you use the OVN-Kubernetes network plugin, you can specify IPv4 and IPv6 networks.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 . The prefix length for an IPv6 block is between 0 and 128 . For example, 10.128.0.0/14 or fd01::/48 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. For an IPv4 network the default value is 23 . For an IPv6 network the default value is 64 . The default value is also the minimum value for IPv6.


Parameter	Description	Values
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p> <p>If you use the OVN-Kubernetes network plugin, you can specify an IP address block for both of the IPv4 and IPv6 address families.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16 - fd02::/112</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16 or fd00::/48.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>

12.4.8.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 12.44. Optional parameters

Parameter	Description	Values
additionalTrustBundle	<p>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</p>	String

Parameter	Description	Values
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled

Parameter	Description	Values
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 and arm64 .	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true

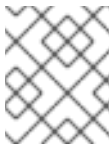
Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 60px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 60px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

12.4.8.2. Sample install-config.yaml file for bare metal

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```
apiVersion: v1
baseDomain: example.com ①
compute: ②
- hyperthreading: Enabled ③
  name: worker
  replicas: 0 ④
controlPlane: ⑤
  hyperthreading: Enabled ⑥
  name: master
  replicas: 3 ⑦
metadata:
  name: test ⑧
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 ⑨
    hostPrefix: 23 ⑩
  networkType: OVNKubernetes ⑪
  serviceNetwork: ⑫
  - 172.30.0.0/16
platform:
  none: {} ⑬
fips: false ⑭
pullSecret: '{"auths":{"<local_registry>":{"auth": "<credentials>","email": "you@example.com"}}}' ⑮
sshKey: 'ssh-ed25519 AAAA...' ⑯
additionalTrustBundle: | ⑰
-----BEGIN CERTIFICATE-----
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
-----END CERTIFICATE-----
imageContentSources: ⑱
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

- ① The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- ② ⑤ The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- ③ ⑥ Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

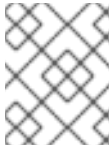
Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for your platform.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 17 Provide the contents of the certificate file that you used for your mirror registry.
- 18 Provide the **imageContentSources** section from the output of the command to mirror the repository.

Additional resources

- See [Load balancing requirements for user-provisioned infrastructure](#) for more information on the API and application ingress load balancing requirements.

12.4.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

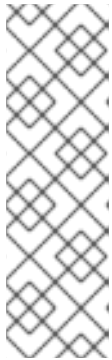


NOTE

For bare metal installations, if you do not assign node IP addresses from the range that is specified in the **networking.machineNetwork[].cidr** field in the **install-config.yaml** file, you must include them in the **proxy.noProxy** field.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then

creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

12.4.8.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
```

```
- name: worker
  platform: {}
  replicas: 0
```



NOTE

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

12.4.9. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

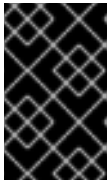
```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.



WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.



IMPORTANT

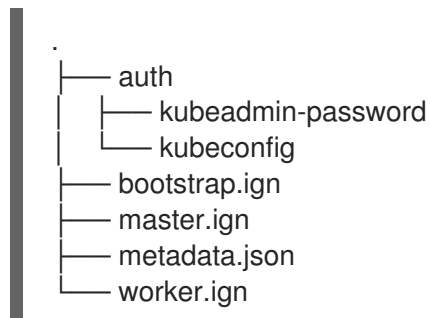
When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:



Additional resources

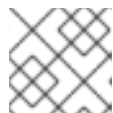
- See [Recovering from expired control plane certificates](#) for more information about recovering kubelet certificates.

12.4.10. Configuring chrony time service

You must set the time server and related settings used by the chrony time service (**chronyd**) by modifying the contents of the **chrony.conf** file and passing those contents to your nodes as a machine config.

Procedure

1. Create a Butane config including the contents of the **chrony.conf** file. For example, to configure chrony on worker nodes, create a **99-worker-chrony.bu** file.



NOTE

See "Creating machine configs with Butane" for information about Butane.

```

variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-chrony 1
  labels:
    machineconfiguration.openshift.io/role: worker 2
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644 3
    overwrite: true
    contents:
      inline: |
        pool 0.rhel.pool.ntp.org iburst 4
        driftfile /var/lib/chrony/drift
        makestep 1.0 3
        rtsync
        logdir /var/log/chrony

```

- 1 2 On control plane nodes, substitute **master** for **worker** in both of these locations.

- 3 Specify an octal value mode for the **mode** field in the machine config file. After creating the file and applying the changes, the **mode** is converted to a decimal value. You can check
 - 4 Specify any valid, reachable time source, such as the one provided by your DHCP server.
2. Use Butane to generate a **MachineConfig** object file, **99-worker-chrony.yaml**, containing the configuration to be delivered to the nodes:

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. Apply the configurations in one of two ways:
 - If the cluster is not running yet, after you generate manifest files, add the **MachineConfig** object file to the **<installation_directory>/openshift** directory, and then continue to create the cluster.
 - If the cluster is already running, apply the file:

```
$ oc apply -f ./99-worker-chrony.yaml
```

12.4.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.



NOTE

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Only RHEL 8 compute machines are supported.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments:** You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the **APPEND** parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special **coreos.inst.*** arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.
- **Ignition configs:** OpenShift Container Platform Ignition config files (***.ign**) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In

special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the **coreos-installer** to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.

- **coreos-installer:** You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the **coreos-installer** command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.



NOTE

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

12.4.11.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

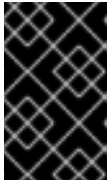
Procedure

1. Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

2. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

3. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left  Speed
  0   0   0   0   0   0   0   0   0  --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

4. Although it is possible to obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS images are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep '\.iso[^\.]'
```

Example output

```
"location": "<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



IMPORTANT

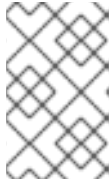
The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:

- Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.



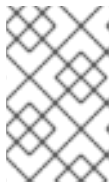
NOTE

It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

- 1** **1** You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.
- 2** The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.



NOTE

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the **/dev/sda** device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

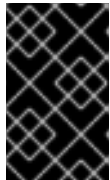
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.4.11.2. Installing RHCOS by using PXE or iPXE booting

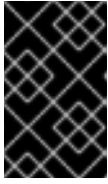
You can use PXE or iPXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE or iPXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total      Spent    Left  Speed
  0   0   0   0   0   0   0   0 --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

3. Although it is possible to obtain the RHCOS **kernel**, **initramfs** and **rootfs** files that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS files are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep -Eo "https.*(kernel-|initramfs-|rootfs-)\w+(\.img)?"
```

Example output

```
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-kernel-
```

```
x86_64"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



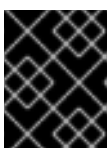
IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** rhcos-<version>-live-kernel-<architecture>
- **initramfs:** rhcos-<version>-live-initramfs.<architecture>.img
- **rootfs:** rhcos-<version>-live-rootfs.<architecture>.img

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.
6. Configure PXE or iPXE installation for the RHCOS images and begin the installation. Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE (**x86_64**):

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

- 1** Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2** If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.

DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.

- 3 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

- For iPXE (**x86_64 + aarch64**):

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **kernel** parameter value is the location of the **kernel** file, the **initrd=main** argument is needed for booting on UEFI systems, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your HTTP server.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **kernel** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

**NOTE**

To network boot the CoreOS **kernel** on **aarch64** architecture, you need to use a version of iPXE build with the **IMAGE_GZIP** option enabled. See [IMAGE_GZIP option in iPXE](#).

- For PXE (with UEFI and Grub as second stage) on **aarch64**:

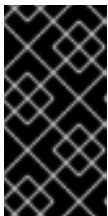
```

menuentry 'Install CoreOS' {
  linux rhcos-<version>-live-kernel-<architecture>
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
  <architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
  initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}

```

- 1** Specify the locations of the RHCOS files that you uploaded to your HTTP/TFTP server. The **kernel** parameter value is the location of the **kernel** file on your TFTP server. The **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file on your HTTP Server.
- 2** If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3** Specify the location of the **initramfs** file that you uploaded to your TFTP server.

7. Monitor the progress of the RHCOS installation on the console of the machine.

**IMPORTANT**

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.
9. Check the console output to verify that Ignition ran.

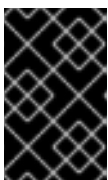
Example command

```

Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied

```

10. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

12.4.11.3. Advanced RHCOS installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
- Customizing a live ISO or PXE boot image

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

12.4.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

Procedure

1. Boot the ISO installer.
2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as **nmcli** or **nmtui**.
3. Run the **coreos-installer** command to install the system, adding the **--copy-network** option to copy networking configuration. For example:

```
$ sudo coreos-installer install --copy-network \  
--ignition-url=http://host/worker.ign /dev/sda
```



IMPORTANT

The **--copy-network** option only copies networking configuration found under **/etc/NetworkManager/system-connections**. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

Additional resources

- See [Getting started with nmcli](#) and [Getting started with nmtui](#) in the RHEL 8 documentation for more information about the **nmcli** and **nmtui** tools.

12.4.11.3.2. Disk partitioning

Disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless you override the default partitioning configuration. During the RHCOS installation, the size of the root file system is increased to use any remaining available space on the target device.



IMPORTANT

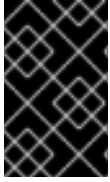
The use of a custom partition scheme on your node might result in OpenShift Container Platform not monitoring or alerting on some node partitions. If you override the default partitioning, see [Understanding OpenShift File System Monitoring \(eviction conditions\)](#) for more information about how OpenShift Container Platform monitors your host file systems.

OpenShift Container Platform monitors the following two filesystem identifiers:

- **nodefs**, which is the filesystem that contains **/var/lib/kubelet**
- **imagefs**, which is the filesystem that contains **/var/lib/containers**

For the default partition scheme, **nodefs** and **imagefs** monitor the same root filesystem, **/**.

To override the default partitioning when installing RHCOS on an OpenShift Container Platform cluster node, you must create separate partitions.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.

Consider a situation where you want to add a separate storage partition for your containers and container images. For example, by mounting **/var/lib/containers** in a separate partition, the kubelet separately monitors **/var/lib/containers** as the **imagefs** directory and the root file system as the **nodefs** directory.



IMPORTANT

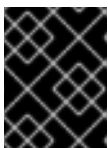
If you have resized your disk size to host a larger file system, consider creating a separate **/var/lib/containers** partition. Consider resizing a disk that has an **xfs** format to reduce CPU time issues caused by a high number of allocation groups.

12.4.11.3.2.1. Creating a separate **/var** partition

In general, you should use the default disk partitioning that is created during the RHCOS installation. However, there are cases where you might want to create a separate partition for a directory that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** directory or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

The use of a separate partition for the **/var** directory or a subdirectory of **/var** also prevents data growth in the partitioned directory from filling up the root file system.

The following procedure sets up a separate **/var** partition by adding a machine config manifest that is wrapped into the Ignition config file for a node type during the preparation phase of an installation.

Procedure

1. On your installation host, change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ openshift-install create manifests --dir <installation_directory>
```

2. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum offset value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no offset value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for compute nodes, if the different instance types do not have the same device name.

3. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. Create the Ignition config files:

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

The files in the **<installation_directory>/manifest** and **<installation_directory>/openshift** directories are wrapped into the Ignition config files, including the file that contains the **98-var-partition** custom **MachineConfig** object.

Next steps

- You can apply the custom disk partitioning by referencing the Ignition config files during the RHCOS installations.

12.4.11.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the **coreos-installer** command that cause the installer to maintain one or more existing partitions. For a PXE installation, you can add **coreos.inst.*** options to the **APPEND** parameter to preserve partitions.

Saved partitions might be data partitions from an existing OpenShift Container Platform system. You can identify the disk partitions you want to keep either by partition label or by number.



NOTE

If you save existing partitions, and those partitions do not leave enough space for RHCOS, the installation will fail without damaging the saved partitions.

Retaining existing partitions during an ISO installation

This example preserves any partition in which the partition label begins with **data** (**data***):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the **coreos-installer** in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
--save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, **coreos-installer** recreates the partition immediately.

Retaining existing partitions during a PXE installation

This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

This **APPEND** option preserves partition 6:

```
coreos.inst.save_partindex=6
```

12.4.11.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config:** Every manual RHCOS installation needs to pass one of the Ignition config files generated by **openshift-installer**, such as **bootstrap.ign**, **master.ign** and **worker.ign**, to carry out the installation.



IMPORTANT

It is not recommended to modify these Ignition config files directly. You can update the manifest files that are wrapped into the Ignition config files, as outlined in examples in the preceding sections.

For PXE installations, you pass the Ignition configs on the **APPEND** line using the **coreos.inst.ignition_url=** option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the **coreos-installer** command line with the **--ignition-url=** option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config:** This type can be created by using the **coreos-installer customize** subcommand and its various options. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be done once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

For PXE or ISO boots, you can create the Ignition config and **APPEND** the **ignition.config.url=** option to identify the location of the Ignition config. You also need to append **ignition.firstboot** **ignition.platform.id=metal** or the **ignition.config.url** option will be ignored.

12.4.11.3.4. Default console configuration

Red Hat Enterprise Linux CoreOS (RHCOS) nodes installed from an OpenShift Container Platform 4.12

boot image use a default console that is meant to accommodate most virtualized and bare metal setups. Different cloud and virtualization platforms may use different default settings depending on the chosen architecture. Bare metal installations use the kernel default settings which typically means the graphical console is the primary console and the serial console is disabled.

The default consoles may not match your specific hardware configuration or you might have specific needs that require you to adjust the default console. For example:

- You want to access the emergency shell on the console for debugging purposes.
- Your cloud platform does not provide interactive access to the graphical console, but provides a serial console.
- You want to enable multiple consoles.

Console configuration is inherited from the boot image. This means that new nodes in existing clusters are unaffected by changes to the default console.

You can configure the console for bare metal installations in the following ways:

- Using **coreos-installer** manually on the command line.
- Using the **coreos-installer iso customize** or **coreos-installer pxe customize** subcommands with the **--dest-console** option to create a custom image that automates the process.



NOTE

For advanced customization, perform console configuration using the **coreos-installer iso** or **coreos-installer pxe** subcommands, and not kernel arguments.

12.4.11.3.5. Enabling the serial console for PXE and ISO installations

By default, the Red Hat Enterprise Linux CoreOS (RHCOS) serial console is disabled and all output is written to the graphical console. You can enable the serial console for an ISO installation and reconfigure the bootloader so that output is sent to both the serial console and the graphical console.

Procedure

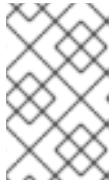
1. Boot the ISO installer.
2. Run the **coreos-installer** command to install the system, adding the **--console** option once to specify the graphical console, and a second time to specify the serial console:

```
$ coreos-installer install \
  --console=tty0 \1
  --console=ttyS0,<options> \2
  --ignition-url=http://host/worker.ign /dev/sda
```

1 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.

2 The desired primary console. In this case the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **11520n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see [Linux kernel serial console](#) documentation.

3. Reboot into the installed system.



NOTE

A similar outcome can be obtained by using the **coreos-installer install --append-karg** option, and specifying the console with **console=**. However, this will only set the console for the kernel and not the bootloader.

To configure a PXE installation, make sure the **coreos.inst.install_dev** kernel command line option is omitted, and use the shell prompt to run **coreos-installer** manually using the above ISO installation procedure.

12.4.11.3.6. Customizing a live RHCOS ISO or PXE install

You can use the live ISO image or PXE environment to install RHCOS by injecting an Ignition config file directly into the image. This creates a customized image that you can use to provision your system.

For an ISO image, the mechanism to do this is the **coreos-installer iso customize** subcommand, which modifies the **.iso** file with your configuration. Similarly, the mechanism for a PXE environment is the **coreos-installer pxe customize** subcommand, which creates a new **initramfs** file that includes your customizations.

The **customize** subcommand is a general purpose tool that can embed other types of customizations as well. The following tasks are examples of some of the more common customizations:

- Inject custom CA certificates for when corporate security policy requires their use.
- Configure network settings without the need for kernel arguments.
- Embed arbitrary preinstall and post-install scripts or binaries.

12.4.11.3.7. Customizing a live RHCOS ISO image

You can customize a live RHCOS ISO image directly with the **coreos-installer iso customize** subcommand. When you boot the ISO image, the customizations are applied automatically.

You can use this feature to configure the ISO image to automatically install RHCOS.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to inject the Ignition config directly into the ISO image:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda 2
```

- 1 The Ignition config file that is generated from the **openshift-installer** installation program.
- 2 When you specify this option, the ISO image automatically runs an installation. Otherwise, the image remains configured for installation, but does not install automatically unless you specify the **coreos.inst.install_dev** kernel argument.

- Optional: To remove the ISO image customizations and return the image to its pristine state, run:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now re-customize the live ISO image or use it in its pristine state.

Applying your customizations affects every subsequent boot of RHCOS.

12.4.11.3.7.1. Modifying a live install ISO image to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

Procedure

- Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
- Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image to enable the serial console to receive output:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/sda \ 4
```

- 1 The location of the Ignition config to install.
- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. In this case, **/dev/sda**. If you omit this option, the ISO image automatically runs the installation program which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.



NOTE

The **--dest-console** option affects the installed system and not the live ISO system. To modify the console for a live ISO system, use the **--live-karg-append** option and specify the console with **console=**.

Your customizations are applied and affect every subsequent boot of the ISO image.

- Optional: To remove the ISO image customizations and return the image to its original state, run the following command:

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

You can now recustomize the live ISO image or use it in its original state.

12.4.11.3.7.2. Modifying a live install ISO image to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



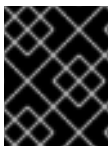
NOTE

Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image for use with a custom CA:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```



IMPORTANT

The **coreos.inst.ignition_url** kernel parameter does not work with the **--ignition-ca** flag. You must use the **--dest-ignition** flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.4.11.3.7.3. Modifying a live install ISO image with customized network settings

You can embed a NetworkManager keyfile into the live ISO image and pass it through to the installed system with the **--network-keyfile** flag of the **customize** subcommand.



WARNING

When creating a connection profile, you must use a **.nmconnection** filename extension in the filename of the connection profile. If you do not use a **.nmconnection** filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.

2. Create a connection profile for a bonded interface. For example, create the **bond0.nmconnection** file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

- Retrieve the RHCOS ISO image from the [RHCOS image mirror](#) page and run the following command to customize the ISO image with your configured networking:

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection
```

Network settings are applied to the live system and are carried over to the destination system.

12.4.11.3.8. Customizing a live RHCOS PXE environment

You can customize a live RHCOS PXE environment directly with the **coreos-installer pxe customize** subcommand. When you boot the PXE environment, the customizations are applied automatically.

You can use this feature to configure the PXE environment to automatically install RHCOS.

Procedure

- Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
- Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new **initramfs** file that contains the customizations from your Ignition config:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/sda \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3
```

- The Ignition config file that is generated from **openshift-installer**.
- When you specify this option, the PXE environment automatically runs an install. Otherwise, the image remains configured for installing, but does not do so automatically unless you specify the **coreos.inst.install_dev** kernel argument.
- Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Applying your customizations affects every subsequent boot of RHCOS.

12.4.11.3.8.1. Modifying a live install PXE environment to enable the serial console

On clusters installed with OpenShift Container Platform 4.12 and above, the serial console is disabled by default and all output is written to the graphical console. You can enable the serial console with the following procedure.

Procedure

- Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
- Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and the Ignition config file, and then run the following command to create a new customized **initramfs** file that enables the serial console to receive output:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--dest-ignition <path> \ 1
--dest-console tty0 \ 2
--dest-console ttyS0,<options> \ 3
--dest-device /dev/sda \ 4
-o rhcos-<version>-custom-initramfs.x86_64.img 5
```

- 1 The location of the Ignition config to install.
- 2 The desired secondary console. In this case, the graphical console. Omitting this option will disable the graphical console.
- 3 The desired primary console. In this case, the serial console. The **options** field defines the baud rate and other settings. A common value for this field is **115200n8**. If no options are provided, the default kernel value of **9600n8** is used. For more information on the format of this option, see the [Linux kernel serial console](#) documentation.
- 4 The specified disk to install to. If you omit this option, the PXE environment automatically runs the installer which will fail unless you also specify the **coreos.inst.install_dev** kernel argument.
- 5 Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.

Your customizations are applied and affect every subsequent boot of the PXE environment.

12.4.11.3.8.2. Modifying a live install PXE environment to use a custom certificate authority

You can provide certificate authority (CA) certificates to Ignition with the **--ignition-ca** flag of the **customize** subcommand. You can use the CA certificates during both the installation boot and when provisioning the installed system.



NOTE

Custom CA certificates affect how Ignition fetches remote resources but they do not affect the certificates installed onto the system.

Procedure

1. Download the **coreos-installer** binary from the [coreos-installer image mirror](#) page.
2. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file for use with a custom CA:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
--ignition-ca cert.pem \
-o rhcos-<version>-custom-initramfs.x86_64.img
```

3. Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present.



IMPORTANT

The `coreos.inst.ignition_url` kernel parameter does not work with the `--ignition-ca` flag. You must use the `--dest-ignition` flag to create a customized image for each cluster.

Applying your custom CA certificate affects every subsequent boot of RHCOS.

12.4.11.3.8.3. Modifying a live install PXE environment with customized network settings

You can embed a NetworkManager keyfile into the live PXE environment and pass it through to the installed system with the `--network-keyfile` flag of the `customize` subcommand.



WARNING

When creating a connection profile, you must use a `.nmconnection` filename extension in the filename of the connection profile. If you do not use a `.nmconnection` filename extension, the cluster will apply the connection profile to the live environment, but it will not apply the configuration when the cluster first boots up the nodes, resulting in a setup that does not work.

Procedure

1. Download the `coreos-installer` binary from the [coreos-installer image mirror](#) page.
2. Create a connection profile for a bonded interface. For example, create the `bond0.nmconnection` file in your local directory with the following content:

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em1.nmconnection** file in your local directory with the following content:

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. Create a connection profile for a secondary interface to add to the bond. For example, create the **bond0-proxy-em2.nmconnection** file in your local directory with the following content:

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. Retrieve the RHCOS **kernel**, **initramfs** and **rootfs** files from the [RHCOS image mirror](#) page and run the following command to create a new customized **initramfs** file that contains your configured networking:

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection \
  -o rhcos-<version>-custom-initramfs.x86_64.img
```

6. Use the customized **initramfs** file in your PXE configuration. Add the **ignition.firstboot** and **ignition.platform.id=metal** kernel arguments if they are not already present. Network settings are applied to the live system and are carried over to the destination system.

12.4.11.3.9. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

12.4.11.3.9.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option.

Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:


```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```


12.4.11.3.9.2. coreos-installer options for ISO and PXE installations

You can install RHCOS by running **coreos-installer install <options> <device>** at the command prompt, after booting into the RHCOS live environment from an ISO image.

The following table shows the subcommands, options, and arguments you can pass to the **coreos-installer** command.


Table 12.45. coreos-installer subcommands, command-line options, and arguments

coreos-installer install subcommand	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer install <options> <device>	Embed an Ignition config in an ISO image.
coreos-installer install subcommand options	
<i>Option</i>	<i>Description</i>
-u, --image-url <url>	Specify the image URL manually.
-f, --image-file <path>	Specify a local image file manually. Used for debugging.

-i, --ignition-file <path>	Embed an Ignition config from a file.
-l, --ignition-url <URL>	Embed an Ignition config from a URL.
--ignition-hash <digest>	Digest type-value of the Ignition config.
-p, --platform <name>	Override the Ignition platform ID for the installed system.
--console <spec>	Set the kernel and bootloader console for the installed system. For more information about the format of <spec> , see the Linux kernel serial console documentation.
--append-karg <arg>...	Append a default kernel argument to the installed system.
--delete-karg <arg>...	Delete a default kernel argument from the installed system.
-n, --copy-network	Copy the network configuration from the install environment. <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>IMPORTANT</p> <p>The --copy-network option only copies networking configuration found under /etc/NetworkManager/system-connections. In particular, it does not copy the system hostname.</p> </div> </div>
--network-dir <path>	For use with -n . Default is /etc/NetworkManager/system-connections/ .
--save-partlabel <lx>..	Save partitions with this label glob.
--save-partindex <id>...	Save partitions with this number or range.
--insecure	Skip RHCOS image signature verification.
--insecure-ignition	Allow Ignition URL without HTTPS or hash.
--architecture <name>	Target CPU architecture. Valid values are x86_64 and aarch64 .
--preserve-on-error	Do not clear partition table on error.
-h, --help	Print help information.

coreos-installer install subcommand argument	
<i>Argument</i>	<i>Description</i>
<device>	The destination device.
coreos-installer ISO subcommands	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer iso customize <options> <ISO_image>	Customize a RHCOS live ISO image.
coreos-installer iso reset <options> <ISO_image>	Restore a RHCOS live ISO image to default settings.
coreos-installer iso ignition remove <options> <ISO_image>	Remove the embedded Ignition config from an ISO image.
coreos-installer ISO customize subcommand options	
<i>Option</i>	<i>Description</i>
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--dest-karg-append <arg>	Add a kernel argument to each boot of the destination system.
--dest-karg-delete <arg>	Delete a kernel argument from each boot of the destination system.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
--post-install <path>	Run the specified script after installation.

--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
--live-karg-append <arg>	Add a kernel argument to each boot of the live environment.
--live-karg-delete <arg>	Delete a kernel argument from each boot of the live environment.
--live-karg-replace <k=o=n>	Replace a kernel argument in each boot of the live environment, in the form key=old=new .
-f, --force	Overwrite an existing Ignition config.
-o, --output <path>	Write the ISO to a new output file.
-h, --help	Print help information.
coreos-installer PXE subcommands	
<i>Subcommand</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
coreos-installer pxe customize <options> <path>	Customize a RHCOS live PXE boot config.
coreos-installer pxe ignition wrap <options>	Wrap an Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <image_name>	Show the wrapped Ignition config in an image.
coreos-installer PXE customize subcommand options	
<i>Option</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.

--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
-o, --output <path>	Write the initramfs to a new output file.  NOTE This option is required for PXE environments.
-h, --help	Print help information.

12.4.11.3.9.3. coreos.inst boot options for ISO or PXE installations

You can automatically invoke **coreos-installer** options at boot time by passing **coreos.inst** boot arguments to the RHCOS live installer. These are provided in addition to the standard boot arguments.

- For ISO installations, the **coreos.inst** options can be added by interrupting the automatic boot at the bootloader menu. You can interrupt the automatic boot by pressing **TAB** while the **RHEL CoreOS (Live)** menu option is highlighted.
- For PXE or iPXE installations, the **coreos.inst** options must be added to the **APPEND** line before the RHCOS live installer is booted.

The following table shows the RHCOS live installer **coreos.inst** boot options for ISO and PXE installations.

Table 12.46. coreos.inst boot options

Argument	Description
coreos.inst.install_dev	Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda , although sda is allowed.

Argument	Description
coreos.inst.ignition_url	Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded. Only HTTP and HTTPS protocols are supported.
coreos.inst.save_partlabel	Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.
coreos.inst.save_partindex	Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.
coreos.inst.insecure	Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.
coreos.inst.image_url	<p>Optional: Download and install the specified RHCOS image.</p> <ul style="list-style-type: none"> ● This argument should not be used in production environments and is intended for debugging purposes only. ● While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install. ● If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform. ● Only HTTP and HTTPS protocols are supported.
coreos.inst.skip_reboot	Optional: The system will not reboot after installing. After the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.

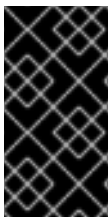
Argument	Description
coreos.inst.platform_id	Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal . This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: coreos.inst.platform_id=vmware.
ignition.config.url	Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from coreos.inst.ignition_url , which is the Ignition config for the installed system.

12.4.11.4. Enabling multipathing with kernel arguments on RHCOS

RHCOS supports multipathing on the primary disk, allowing stronger resilience to hardware failure to achieve higher host availability.

You can enable multipathing at installation time for nodes that were provisioned in OpenShift Container Platform 4.8 or later. While postinstallation support is available by activating multipathing via the machine config, enabling multipathing during installation is recommended.

In setups where any I/O to non-optimized paths results in I/O system errors, you must enable multipathing at installation time.



IMPORTANT

On IBM Z and IBM® LinuxONE, you can enable multipathing only if you configured your cluster for it during installation. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process" in *Installing a cluster with z/VM on IBM Z and IBM® LinuxONE*.

The following procedure enables multipath at installation time and appends kernel arguments to the **coreos-installer install** command so that the installed system itself will use multipath beginning from the first boot.



NOTE

OpenShift Container Platform does not support enabling multipathing as a day-2 activity on nodes that have been upgraded from 4.6 or earlier.

Procedure

1. To enable multipath and start the **multipathd** daemon, run the following command on the installation host:

```
$ mpathconf --enable && systemctl start multipathd.service
```

- Optional: If booting the PXE or ISO, you can instead enable multipath by adding **rd.multipath=default** from the kernel command line.
2. Append the kernel arguments by invoking the **coreos-installer** program:
 - If there is only one multipath device connected to the machine, it should be available at path **/dev/mapper/mpatha**. For example:

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1 Indicates the path of the single multipathed device.

- If there are multiple multipath devices connected to the machine, or to be more explicit, instead of using **/dev/mapper/mpatha**, it is recommended to use the World Wide Name (WWN) symlink available in **/dev/disk/by-id**. For example:

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1 Indicates the WWN ID of the target multipathed device. For example, **0xx194e957fcedb4841**.

This symlink can also be used as the **coreos.inst.install_dev** kernel argument when using special **coreos.inst.*** arguments to direct the live installer. For more information, see "Installing RHCOS and starting the OpenShift Container Platform bootstrap process".

3. Check that the kernel arguments worked by going to one of the worker nodes and listing the kernel command line arguments (in **/proc/cmdline** on the host):

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

Example output

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

You should see the added kernel arguments.

12.4.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the

persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

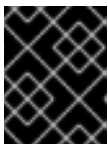
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

Additional resources

- See [Monitoring installation progress](#) for more information about monitoring the installation logs and retrieving diagnostic data if installation issues arise.

12.4.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The

kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

12.4.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

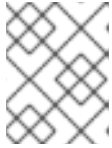
1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE  VERSION
master-0  Ready    master   63m  v1.25.0
master-1  Ready    master   63m  v1.25.0
master-2  Ready    master   64m  v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

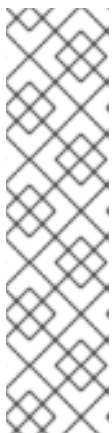
```
NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n}}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

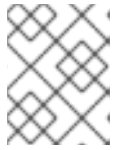
Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
```

```

master-2 Ready   master 74m v1.25.0
worker-0 Ready   worker 11m v1.25.0
worker-1 Ready   worker 11m v1.25.0

```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

12.4.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m

node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

Additional resources

- See [Gathering logs from a failed installation](#) for details about gathering data in the event of a failed OpenShift Container Platform installation.
- See [Troubleshooting Operator issues](#) for steps to check Operator pod health across the cluster and gather Operator logs for diagnosis.

12.4.15.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

12.4.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

12.4.15.2.1. Changing the image registry's management state

To start the image registry, you must change the Image Registry Operator configuration's **managementState** from **Removed** to **Managed**.

Procedure

- Change **managementState** Image Registry Operator configuration from **Removed** to **Managed**. For example:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"managementState": "Managed"}}'
```

12.4.15.2.2. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

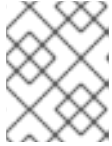
When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```

**NOTE**

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

12.4.15.2.3. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```


**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

12.4.15.2.4. Configuring block registry storage for bare metal

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes, or block persistent volumes, are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

If you choose to use a block storage volume with the image registry, you must use a filesystem persistent volume claim (PVC).

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only one (**1**) replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
    - ReadWriteOnce 3
```

```
resources:
  requests:
    storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

12.4.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME SINCE	VERSION	AVAILABLE	PROGRESSING	DEGRADED	
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

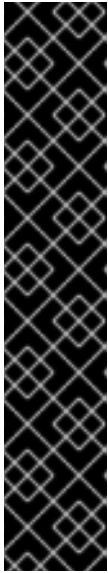
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running   0    5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

4. Register your cluster on the [Cluster registration](#) page.

12.4.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

12.4.18. Next steps

- [Validating an installation.](#)
- [Customize your cluster.](#)
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#) .
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)

CHAPTER 13. INSTALLING ON-PREMISE WITH ASSISTED INSTALLER

13.1. INSTALLING AN ON-PREMISE CLUSTER USING THE ASSISTED INSTALLER

You can install OpenShift Container Platform on on-premise hardware or on-premise VMs using the Assisted Installer. Installing OpenShift Container Platform using the Assisted Installer supports both x86_64 and AArch64 CPU architectures.

13.1.1. Using the Assisted Installer

The [Assisted Installer](#) is a user-friendly installation solution offered on the [Red Hat Hybrid Cloud Console](#). The Assisted Installer supports the various deployment platforms with a focus on bare metal, Nutanix, and vSphere infrastructures.

The Assisted Installer provides installation functionality as a service. This software-as-a-service (SaaS) approach has the following advantages:

- **Web user interface:** The web user interface performs cluster installation without the user having to create the installation configuration files manually.
- **No bootstrap node:** A bootstrap node is not required when installing with the Assisted Installer. The bootstrapping process executes on a node within the cluster.
- **Hosting:** The Assisted Installer hosts:
 - Ignition files
 - The installation configuration
 - A discovery ISO
 - The installer
- **Streamlined installation workflow:** Deployment does not require in-depth knowledge of OpenShift Container Platform. The Assisted Installer provides reasonable defaults and provides the installer as a service, which:
 - Eliminates the need to install and run the OpenShift Container Platform installer locally.
 - Ensures the latest version of the installer up to the latest tested z-stream releases. Older versions remain available, if needed.
 - Enables building automation by using the API without the need to run the OpenShift Container Platform installer locally.
- **Advanced networking:** The Assisted Installer supports IPv4 networking with SDN and OVN, IPv6 and dual stack networking with OVN only, NMState-based static IP addressing, and an HTTP/S proxy. OVN is the default Container Network Interface (CNI) for OpenShift Container Platform 4.12 and later, but you can use SDN.
- **Preinstallation validation:** The Assisted Installer validates the configuration before installation to ensure a high probability of success. The validation process includes the following checks:

- Ensuring network connectivity
 - Ensuring sufficient network bandwidth
 - Ensuring connectivity to the registry
 - Ensuring time synchronization between cluster nodes
 - Verifying that the cluster nodes meet the minimum hardware requirements
 - Validating the installation configuration parameters
- **REST API:** The Assisted Installer has a REST API, enabling automation.

The Assisted Installer supports installing OpenShift Container Platform on premises in a connected environment, including with an optional HTTP/S proxy. It can install the following:

- Highly available OpenShift Container Platform or single-node OpenShift (SNO)
- OpenShift Container Platform on bare metal, Nutanix, or vSphere with full platform integration, or other virtualization platforms without integration
- Optionally OpenShift Virtualization and OpenShift Data Foundation (formerly OpenShift Container Storage)

The user interface provides an intuitive interactive workflow where automation does not exist or is not required. Users may also automate installations using the REST API.

See the [Assisted Installer for OpenShift Container Platform](#) documentation for details.

13.1.2. API support for the Assisted Installer

Supported APIs for the Assisted Installer are stable for a minimum of three months from the announcement of deprecation.

CHAPTER 14. INSTALLING AN ON-PREMISE CLUSTER WITH THE AGENT-BASED INSTALLER

14.1. PREPARING TO INSTALL WITH THE AGENT-BASED INSTALLER

14.1.1. About the Agent-based Installer

The Agent-based installation method provides the flexibility to boot your on-premises servers in any way that you choose. It combines the ease of use of the Assisted Installation service with the ability to run offline, including in air-gapped environments. Agent-based installation is a subcommand of the OpenShift Container Platform installer. It generates a bootable ISO image containing all of the information required to deploy an OpenShift Container Platform cluster, with an available release image.

The configuration is in the same format as for the installer-provisioned infrastructure and user-provisioned infrastructure installation methods. The Agent-based Installer can also optionally generate or accept Zero Touch Provisioning (ZTP) custom resources. ZTP allows you to provision new edge sites with declarative configurations of bare-metal equipment.

14.1.2. Understanding Agent-based Installer

As an OpenShift Container Platform user, you can leverage the advantages of the Assisted Installer hosted service in disconnected environments.

The Agent-based installation comprises a bootable ISO that contains the Assisted discovery agent and the Assisted Service. Both are required to perform the cluster installation, but the latter runs on only one of the hosts.

The **openshift-install agent create image** subcommand generates an ephemeral ISO based on the inputs that you provide. You can choose to provide inputs through the following manifests:

Preferred:

- **install-config.yaml**
- **agent-config.yaml**

or

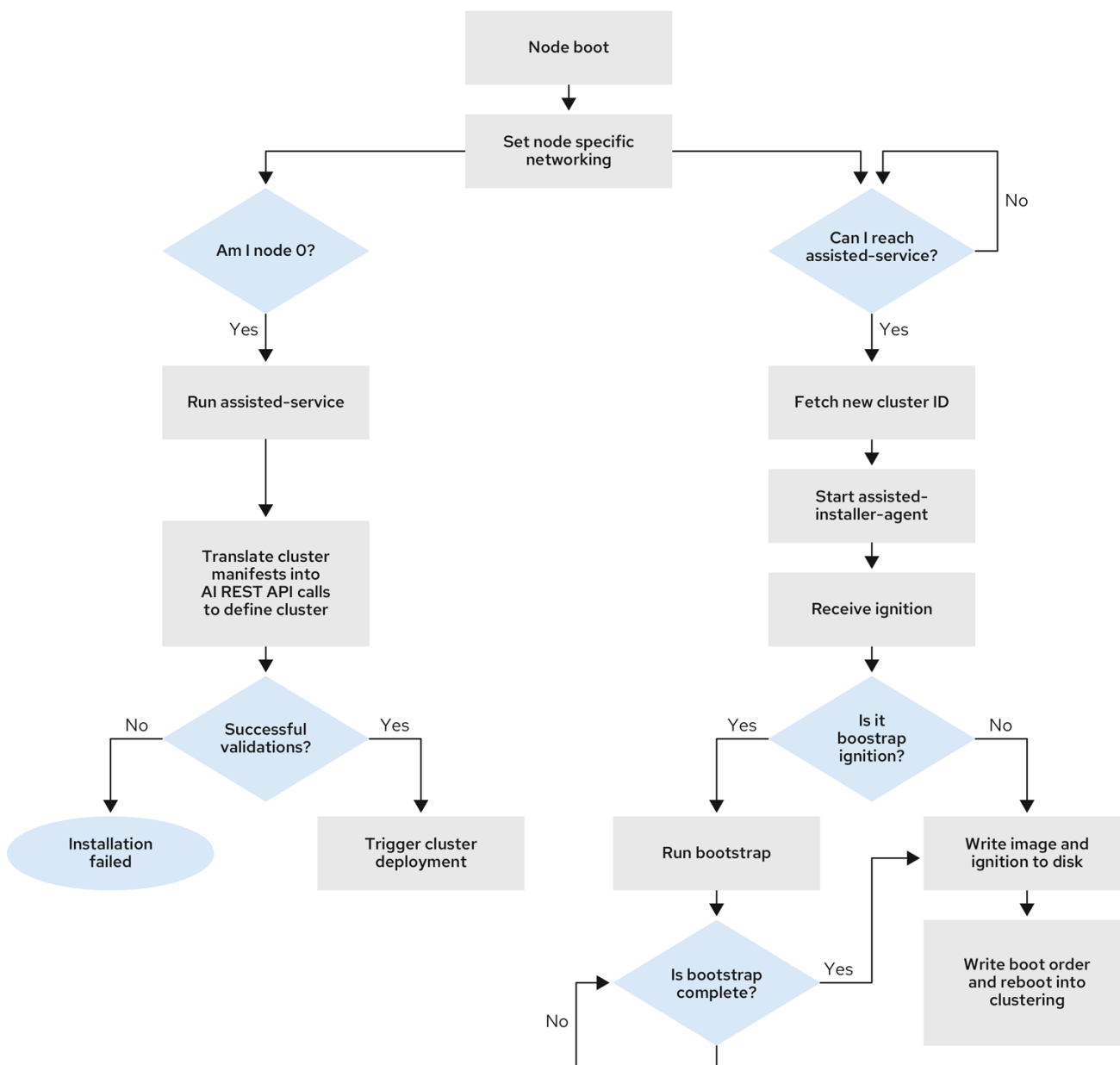
Optional: ZTP manifests

- **cluster-manifests/cluster-deployment.yaml**
- **cluster-manifests/agent-cluster-install.yaml**
- **cluster-manifests/pull-secret.yaml**
- **cluster-manifests/infraenv.yaml**
- **cluster-manifests/cluster-image-set.yaml**
- **cluster-manifests/nmstateconfig.yaml**
- **mirror/registries.conf**
- **mirror/ca-bundle.crt**

14.1.2.1. Agent-based Installer workflow

One of the control plane hosts runs the Assisted Service at the start of the boot process and eventually becomes the bootstrap host. This node is called the **rendezvous host** (node 0). The Assisted Service ensures that all the hosts meet the requirements and triggers an OpenShift Container Platform cluster deployment. All the nodes have the Red Hat Enterprise Linux CoreOS (RHCOS) image written to the disk. The non-bootstrap nodes reboot and initiate a cluster deployment. Once the nodes are rebooted, the rendezvous host reboots and joins the cluster. The bootstrapping is complete and the cluster is deployed.

Figure 14.1. Node installation workflow



281_OpenShift_1022

You can install a disconnected OpenShift Container Platform cluster through the **openshift-install agent create image** subcommand for the following topologies:

- **A single-node OpenShift Container Platform cluster (SNO)** A node that is both a master and worker.

- **A three-node OpenShift Container Platform cluster:** A compact cluster that has three master nodes that are also worker nodes.
- **Highly available OpenShift Container Platform cluster (HA)** Three master nodes with any number of worker nodes.

14.1.2.2. Recommended resources for topologies

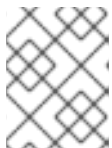
Recommended cluster resources for the following topologies:

Table 14.1. Recommended cluster resources

Topology	Number of master nodes	Number of worker nodes	vCPU	Memory	Storage
Single-node cluster	1	0	8 vCPU cores	16GB of RAM	120GB
Compact cluster	3	0 or 1	8 vCPU cores	16GB of RAM	120GB
HA cluster	3	2 and above	8 vCPU cores	16GB of RAM	120GB

The following platforms are supported:

- **baremetal**
- **vsphere**
- **none**

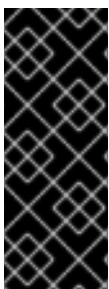


NOTE

The **none** option is supported for only single-node OpenShift clusters with an **OVNKubernetes** network type.

14.1.3. About FIPS compliance

For many OpenShift Container Platform customers, regulatory readiness, or compliance, on some level is required before any systems can be put into production. That regulatory readiness can be imposed by national standards, industry standards or the organization's corporate governance framework. Federal Information Processing Standards (FIPS) compliance is one of the most critical components required in highly secure environments to ensure that only supported cryptographic technologies are allowed on nodes.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

14.1.4. Configuring FIPS through the Agent-based Installer

During a cluster deployment, the Federal Information Processing Standards (FIPS) change is applied when the Red Hat Enterprise Linux CoreOS (RHCOS) machines are deployed in your cluster. For Red Hat Enterprise Linux (RHEL) machines, you must enable FIPS mode when you install the operating system on the machines that you plan to use as worker machines.

You can enable FIPS mode through the preferred method of **install-config.yaml** and **agent-config.yaml**:

1. You must set value of the **fips** field to **True** in the **install-config.yaml** file:

Sample install-config.yaml file

```
apiVersion: v1
baseDomain: test.example.com
metadata:
  name: sno-cluster
fips: True
```

2. Optional: If you are using the ZTP manifests, you must set the value of **fips** as **True** in the **Agent-install.openshift.io/install-config-overrides** field in the **agent-cluster-install.yaml** file:

Sample agent-cluster-install.yaml file

```
apiVersion: extensions.hive.openshift.io/v1beta1
kind: AgentClusterInstall
metadata:
  annotations:
    agent-install.openshift.io/install-config-overrides: '{"fips": True}'
  name: sno-cluster
  namespace: sno-cluster-test
```

Additional resources

- [OpenShift Security Guide Book](#)
- [Support for FIPS cryptography](#)

14.1.5. About networking

The **rendezvous IP** must be known at the time of generating the agent ISO, so that during the initial boot all the hosts can check in to the assisted service. If the IP addresses are assigned using a Dynamic Host Configuration Protocol (DHCP) server, then the **rendezvousIP** field must be set to an IP address of one of the hosts that will become part of the deployed control plane. In an environment without a DHCP server, you can define IP addresses statically.

In addition to static IP addresses, you can apply any network configuration that is in NMState format. This includes VLANs and NIC bonds.

14.1.5.1. DHCP

Preferred method: install-config.yaml and agent-config.yaml

You must specify the value for the **rendezvousIP** field. The **networkConfig** fields can be left blank:

Sample agent-config.yaml file

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 1
```

- 1** The IP address for the rendezvous host.

14.1.5.2. Static networking

- a. Preferred method: **install-config.yaml** and **agent-config.yaml**

Sample agent-config.yaml file

```
cat > agent-config.yaml << EOF
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 1
hosts:
  - hostname: master-0
  interfaces:
    - name: eno1
      macAddress: 00:ef:44:21:e6:a5 2
  networkConfig:
    interfaces:
      - name: eno1
        type: ethernet
        state: up
        mac-address: 00:ef:44:21:e6:a5
        ipv4:
          enabled: true
          address:
            - ip: 192.168.111.80 3
              prefix-length: 23 4
            dhcp: false
        dns-resolver:
          config:
            server:
              - 192.168.111.1 5
        routes:
          config:
            - destination: 0.0.0.0/0
              next-hop-address: 192.168.111.1 6
              next-hop-interface: eno1
              table-id: 254
EOF
```

- 1** If a value is not specified for the **rendezvousIP** field, one address will be chosen from the static IP addresses specified in the **networkConfig** fields.

- 2 The MAC address of an interface on the host, used to determine which host to apply the configuration to.
- 3 The static IP address of the target bare metal host.
- 4 The static IP address's subnet prefix for the target bare metal host.
- 5 The DNS server for the target bare metal host.
- 6 Next hop address for the node traffic. This must be in the same subnet as the IP address set for the specified interface.

b. Optional method: ZTP manifests

The optional method of the ZTP custom resources comprises 6 custom resources; you can configure static IPs in the **nmstateconfig.yaml** file.

```

apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2 1
              prefix-length: 23 2
          dhcp: false
        dns-resolver:
          config:
            server:
              - 192.168.122.1 3
        routes:
          config:
            - destination: 0.0.0.0/0
              next-hop-address: 192.168.122.1 4
              next-hop-interface: eth0
              table-id: 254
    interfaces:
      - name: eth0
        macAddress: 52:54:01:aa:aa:a1 5

```

- 1 The static IP address of the target bare metal host.
- 2 The static IP address's subnet prefix for the target bare metal host.

- 3 The DNS server for the target bare metal host.
- 4 Next hop address for the node traffic. This must be in the same subnet as the IP address set for the specified interface.
- 5 The MAC address of an interface on the host, used to determine which host to apply the configuration to.

The rendezvous IP is chosen from the static IP addresses specified in the **config** fields.

14.1.6. Example: Bonds and VLAN interface node network configuration

The following **agent-config.yaml** file is an example of a manifest for bond and VLAN interfaces.

```

apiVersion: v1alpha1
kind: AgentConfig
rendezvousIP: 10.10.10.14
hosts:
- hostname: master0
  role: master
  interfaces:
  - name: enp0s4
    macAddress: 00:21:50:90:c0:10
  - name: enp0s5
    macAddress: 00:21:50:90:c0:20
networkConfig:
  interfaces:
  - name: bond0.300 1
    type: vlan 2
    state: up
    vlan:
      base-iface: bond0
      id: 300
    ipv4:
      enabled: true
      address:
        - ip: 10.10.10.14
          prefix-length: 24
      dhcp: false
  - name: bond0 3
    type: bond 4
    state: up
    mac-address: 00:21:50:90:c0:10 5
    ipv4:
      enabled: false
    ipv6:
      enabled: false
    link-aggregation:
      mode: active-backup 6
      options:
        miimon: "150" 7
    port:
      - enp0s4
      - enp0s5

```

```

dns-resolver: 8
  config:
    server:
      - 10.10.10.11
      - 10.10.10.12
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 10.10.10.10 9
          next-hop-interface: bond0.300 10
          table-id: 254

```

- 1 3 Name of the interface.
- 2 The type of interface. This example creates a VLAN.
- 4 The type of interface. This example creates a bond.
- 5 The mac address of the interface.
- 6 The **mode** attribute specifies the bonding mode.
- 7 Specifies the MII link monitoring frequency in milliseconds. This example inspects the bond link every 150 milliseconds.
- 8 Optional: Specifies the search and server settings for the DNS server.
- 9 Next hop address for the node traffic. This must be in the same subnet as the IP address set for the specified interface.
- 10 Next hop interface for the node traffic.

14.1.7. Sample install-config.yaml file for bare metal

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  - name: worker
    replicas: 0 3
controlPlane: 4
  name: master
  replicas: 1 5
metadata:
  name: sno-cluster 6
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14 7
      hostPrefix: 23 8
  networkType: OVNKubernetes 9
  serviceNetwork: 10

```

```

- 172.30.0.0/16
platform:
  none: {} 11
fips: false 12
pullSecret: '{"auths": ...}' 13
sshKey: 'ssh-ed25519 AAAA...' 14

```

- 1** The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2** **4** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3** This parameter controls the number of compute machines that the Agent-based installation waits to discover before triggering the installation process. It is the number of compute machines that must be booted with the generated ISO.



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

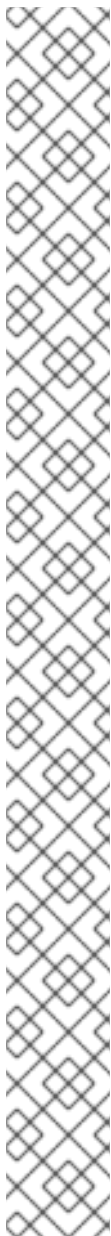
- 5** The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 6** The cluster name that you specified in your DNS records.
- 7** A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 8** The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 9** The cluster network plugin to install. The supported values are **OVNKubernetes** (default value) and **OpenShiftSDN**.
- 10** The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 11** You must set the platform to **none** for a single-node cluster. You can set the platform to either **vsphere** or **baremetal** for multi-node clusters.



NOTE

If you set the platform to **vsphere** or **baremetal**, you can configure IP address endpoints for cluster nodes in three ways:

- IPv4
- IPv6
- IPv4 and IPv6 in parallel (dual-stack)

Example of dual-stack networking

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
      - 192.168.11.3
      - 2001:DB8::4
    ingressVIPs:
      - 192.168.11.4
      - 2001:DB8::5
```

- 12** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 13** This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 14** The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

14.1.8. Validation checks before agent ISO creation

The Agent-based Installer performs validation checks on user defined YAML files before the ISO is created. Once the validations are successful, the agent ISO is created.

install-config.yaml

- **baremetal**, **vsphere** and **none** platforms are supported.
- If **none** is used as a platform, the number of control plane replicas must be **1** and the total number of worker replicas must be **0**.
- The **networkType** parameter must be **OVNKubernetes** in the case of **none** platform.
- **apiVIPs** and **ingressVIPs** parameters must be set for bare metal and vSphere platforms.
- Some host-specific fields in the bare metal platform configuration that have equivalents in **agent-config.yaml** file are ignored. A warning message is logged if these fields are set.

agent-config.yaml

- Each interface must have a defined MAC address. Additionally, all interfaces must have a different MAC address.
- At least one interface must be defined for each host.
- World Wide Name (WWN) vendor extensions are not supported in root device hints.
- The **role** parameter in the **host** object must have a value of either **master** or **worker**.

14.1.8.1. ZTP manifests

agent-cluster-install.yaml

- For IPv6, the only supported value for the **networkType** parameter is **OVNKubernetes**. The **OpenshiftSDN** value can be used only for IPv4.

cluster-image-set.yaml

- The **ReleaseImage** parameter must match the release defined in the installer.

14.1.9. About root device hints

The **rootDeviceHints** parameter enables the installer to provision the Red Hat Enterprise Linux CoreOS (RHCOS) image to a particular device. The installer examines the devices in the order it discovers them, and compares the discovered values with the hint values. The installer uses the first discovered device

that matches the hint value. The configuration can combine multiple hints, but a device must match all hints for the installer to select it.

Table 14.2. Subfields

Subfield	Description
deviceName	A string containing a Linux device name like /dev/vda . The hint must match the actual value exactly.
hctl	A string containing a SCSI bus address like 0:0:0:0 . The hint must match the actual value exactly.
model	A string containing a vendor-specific device identifier. The hint can be a substring of the actual value.
vendor	A string containing the name of the vendor or manufacturer of the device. The hint can be a substring of the actual value.
serialNumber	A string containing the device serial number. The hint must match the actual value exactly.
minSizeGigabytes	An integer representing the minimum size of the device in gigabytes.
wwn	A string containing the unique storage identifier. The hint must match the actual value exactly.
rotational	A boolean indicating whether the device should be a rotating disk (true) or not (false).

Example usage

```
- name: master-0
  role: master
  rootDeviceHints:
    deviceName: "/dev/sda"
```

14.1.10. Next steps

- [Installing a cluster with the Agent-based Installer](#)

14.2. UNDERSTANDING DISCONNECTED INSTALLATION MIRRORING

You can use a mirror registry for disconnected installations and to ensure that your clusters only use container images that satisfy your organization's controls on external content. Before you install a cluster on infrastructure that you provision in a disconnected environment, you must mirror the required

container images into that environment. To mirror container images, you must have a registry for mirroring.

14.2.1. Mirroring images for a disconnected installation through the Agent-based Installer

You can use one of the following procedures to mirror your OpenShift Container Platform image repository to your mirror registry:

- [Mirroring images for a disconnected installation](#)
- [Mirroring images for a disconnected installation using the oc-mirror plugin](#)

14.2.2. About mirroring the OpenShift Container Platform image repository for a disconnected registry

To use mirror images for a disconnected installation with the Agent-based Installer, you must modify the **install-config.yaml** file.

You can mirror the release image by using the output of either the **oc adm release mirror** or **oc mirror** command. This is dependent on which command you used to set up the mirror registry.

The following example shows the output of the **oc adm release mirror** command.

```
$ oc adm release mirror
```

Example output

To use the new mirrored repository to install, add the following section to the **install-config.yaml**:

```
imageContentSources:  
  
  mirrors:  
  virthost.ostest.test.metalkube.org:5000/localimages/local-release-image  
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev  
  mirrors:  
  virthost.ostest.test.metalkube.org:5000/localimages/local-release-image  
  source: registry.ci.openshift.org/ocp/release
```

The following example shows part of the **imageContentSourcePolicy.yaml** file generated by the **oc-mirror** plugin. The file can be found in the results directory, for example **oc-mirror-workspace/results-1682697932/**.

Example imageContentSourcePolicy.yaml file

```
spec:  
  repositoryDigestMirrors:  
  - mirrors:  
    - virthost.ostest.test.metalkube.org:5000/openshift/release  
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev  
  - mirrors:  
    - virthost.ostest.test.metalkube.org:5000/openshift/release-images  
    source: quay.io/openshift-release-dev/ocp-release
```

14.2.2.1. Configuring the Agent-based Installer to use mirrored images

You must use the output of either the **oc adm release mirror** command or the `oc-mirror` plugin to configure the Agent-based Installer to use mirrored images.

Procedure

1. If you used the `oc-mirror` plugin to mirror your release images:
 - a. Open the **imageContentSourcePolicy.yaml** located in the results directory, for example **oc-mirror-workspace/results-1682697932/**.
 - b. Copy the text in the **repositoryDigestMirrors** section of the yaml file.
2. If you used the **oc adm release mirror** command to mirror your release images:
 - Copy the text in the **imageContentSources** section of the command output.
3. Paste the copied text into the **imageContentSources** field of the **install-config.yaml** file.
4. Add the certificate file used for the mirror registry to the **additionalTrustBundle** field of the yaml file.



IMPORTANT

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

Example install-config.yaml file

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----END CERTIFICATE-----
```

5. If you are using `{ztp}` manifests: add the **registries.conf** and **ca-bundle.crt** files to the **mirror** path to add the mirror configuration in the agent ISO image.



NOTE

You can create the **registries.conf** file from the output of either the **oc adm release mirror** command or the **oc mirror** plugin. The format of the `/etc/containers/registries.conf` file has changed. It is now version 2 and in TOML format.

Example registries.conf file

```
[[registry]]
location = "registry.ci.openshift.org/ocp/release" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"
```

```
[[registry]]
location = "quay.io/openshift-release-dev/ocp-v4.0-art-dev" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"
```

14.3. INSTALLING AN OPENSIFT CONTAINER PLATFORM CLUSTER WITH THE AGENT-BASED INSTALLER

14.3.1. Prerequisites

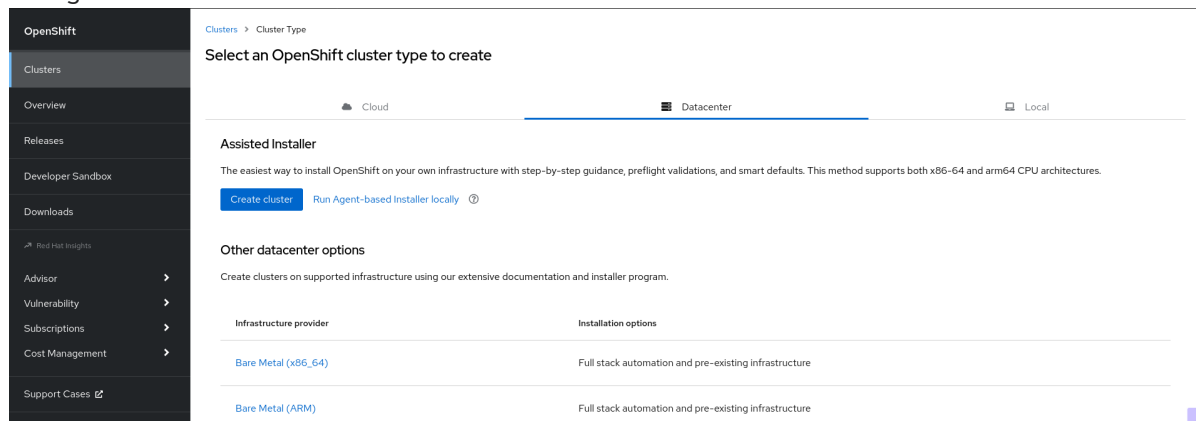
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall or proxy, you [configured it to allow the sites](#) that your cluster requires access to.

14.3.2. Installing OpenShift Container Platform with the Agent-based Installer

The following procedure deploys a single-node OpenShift Container Platform in a disconnected environment. You can use this procedure as a basis and modify according to your requirements.

Procedure

1. Log in to the OpenShift Container Platform web console using your login credentials.
2. Navigate to [Datacenter](#).



3. Click [Run Agent-based Installer locally](#). You are directed to the [Install OpenShift Container Platform on Bare Metal locally with Agent](#) page.
4. Optional: Alternatively, you can also click [Bare Metal \(x86_64\)](#) on the [Select an OpenShift Container Platform cluster type to create](#) page. You are directed to the [Create an OpenShift Container Platform Cluster: Bare Metal](#) page. Then, select [Local Agent-based](#) to go to the [Install OpenShift Container Platform on Bare Metal locally with Agent](#) page.

Clusters > Cluster Type > Bare Metal

Create an OpenShift Cluster: Bare Metal

Select the installation type that best fits your needs.

Interactive

★ Recommended Web-based

Runs Assisted Installer with standard configuration settings to create your cluster.

- ✓ Preflight validations
- ✓ Smart defaults
- ✓ For connected networks

[Learn more about interactive](#)

Local Agent-based

CLI-based

Runs Assisted Installer securely and locally to create your cluster.

- ✓ Installable ISO
- ✓ Preflight validations
- ✓ For air-gapped/restricted networks

[Learn more about local agent-based](#)

Automated

CLI-based

Auto-provision your infrastructure with minimal configuration to create your cluster.

- ✓ Installer Provisioned Infrastructure
- ✓ Hosts controlled with baseboard management controller (BMC)
- ✓ For air-gapped/restricted networks

[Learn more about automated](#)

Full control

CLI-based

Make all of the decisions when you create your cluster.

- ✓ User Provisioned Infrastructure
- ✓ Highly customizable
- ✓ For air-gapped/restricted networks

[Learn more about full control](#)

5. Select the operating system and architecture.
6. Click **Download Installer** to download and extract the install program.
7. You can either download or copy the pull secret by clicking on **Download pull secret** or **Copy pull secret**.
8. Click **Download command-line tools** and place the **openshift-install** binary in a directory that is on your **PATH**.
9. Install **nmstate** dependency by running the following command:

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

10. Place the **openshift-install** binary in a directory that is on your PATH.
11. Create a directory to store the install configuration by running the following command:

```
$ mkdir ~/<directory_name>
```



NOTE

This is the preferred method for the Agent-based installation. Using ZTP manifests is optional.

12. Create the **install-config.yaml** file:

```
$ cat << EOF > ./my-cluster/install-config.yaml
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster 1
```

```
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 192.168.111.0/16
  networkType: OVNKubernetes 2
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
pullSecret: '<pull_secret>' 3
sshKey: |
  <ssh_pub_key> 4
EOF
```

- 1** Required.
- 2** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 3** Enter your pull secret.
- 4** Enter your ssh public key.

NOTE

If you set the platform to **vSphere** or **baremetal**, you can configure IP address endpoints for cluster nodes in three ways:

- IPv4
- IPv6
- IPv4 and IPv6 in parallel (dual-stack)

Example of dual-stack networking

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
      - 192.168.11.3
      - 2001:DB8::4
    ingressVIPs:
      - 192.168.11.4
      - 2001:DB8::5
```

IPv6 is supported only on bare metal platforms.

13. Create the **agent-config.yaml** file:

```
$ cat > agent-config.yaml << EOF
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 1
hosts: 2
  - hostname: master-0 3
  interfaces:
    - name: eno1
      macAddress: 00:ef:44:21:e6:a5
  rootDeviceHints: 4
    deviceName: /dev/sdb
  networkConfig: 5
    interfaces:
      - name: eno1
```

```

type: ethernet
state: up
mac-address: 00:ef:44:21:e6:a5
ipv4:
  enabled: true
  address:
    - ip: 192.168.111.80
      prefix-length: 23
  dhcp: false
dns-resolver:
  config:
    server:
      - 192.168.111.1
routes:
  config:
    - destination: 0.0.0.0/0
      next-hop-address: 192.168.111.2
      next-hop-interface: eno1
      table-id: 254

```

EOF

- 1 This IP address is used to determine which node performs the bootstrapping process as well as running the **assisted-service** component. You must provide the rendezvous IP address when you do not specify at least one host's IP address in the **networkConfig** parameter. If this address is not provided, one IP address is selected from the provided hosts' **networkConfig**.
- 2 Host configuration is optional. The number of hosts defined must not exceed the total number of hosts defined in the **install-config.yaml** file, which is the sum of the values of the **compute.replicas** and **controlPlane.replicas** parameters.
- 3 The optional **hostname** parameter overrides the hostname obtained from either the Dynamic Host Configuration Protocol (DHCP) or a reverse DNS lookup. Each host must have a unique hostname supplied by one of these methods.
- 4 The **rootDeviceHints** parameter enables provisioning of the Red Hat Enterprise Linux CoreOS (RHCOS) image to a particular device. It examines the devices in the order it discovers them, and compares the discovered values with the hint values. It uses the first discovered device that matches the hint value.
- 5 Set this optional parameter to configure the network interface of a host in NMState format.

14. Create the agent image by running the following command:

```
$ openshift-install --dir <install_directory> agent create image
```



NOTE

Red Hat Enterprise Linux CoreOS (RHCOS) supports multipathing on the primary disk, allowing stronger resilience to hardware failure to achieve higher host availability. Multipathing is enabled by default in the agent ISO image, with a default **/etc/multipath.conf** configuration.

15. Boot the **agent.x86_64.iso** image on the bare metal machines.
16. Optional: To know when the bootstrap host (which is the rendezvous host) reboots, run the following command:

```
$ ./openshift-install --dir <install_directory> agent wait-for bootstrap-complete \ 1
--log-level=info 2
```

1 For **<install_directory>**, specify the path to the directory where the agent ISO was generated.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
.....
.....
INFO Bootstrap configMap status is complete
INFO cluster bootstrap is complete
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

17. To track the progress and verify successful installation, run the following command:

```
$ openshift-install --dir <install_directory> agent wait-for install-complete 1
```

1 For **<install_directory>** directory, specify the path to the directory where the agent ISO was generated.

Example output

```
.....
.....
INFO Cluster is installed
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
INFO   export KUBECONFIG=/home/core/installer/auth/kubeconfig
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.sno-
cluster.test.example.com
```

**NOTE**

If you are using the optional method of ZTP manifests, you can configure IP address endpoints for cluster nodes through the **AgentClusterInstall.yaml** file in three ways:

- IPv4
- IPv6
- IPv4 and IPv6 in parallel (dual-stack)

Example of dual-stack networking

```
apiVIP: 192.168.11.3
ingressVIP: 192.168.11.4
clusterDeploymentRef:
  name: mycluster
imageSetRef:
  name: openshift-4.12
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
```

IPv6 is supported only on bare metal platforms.

Additional resources

- See [Deploying with dual-stack networking](#) .
- See [Configuring the install-config yaml file](#) .
- See [Configuring a three-node cluster](#) to deploy three-node clusters in bare metal environments.
- See [About root device hints](#) .
- See [NMState state examples](#) .

14.3.3. Gathering log data from a failed Agent-based installation

Use the following procedure to gather log data about a failed Agent-based installation to provide for a support case.

Procedure

1. Run the following command and collect the output:

```
$ ./openshift-install --dir <install_directory> agent wait-for bootstrap-complete --log-level=debug
```

Example error message

```
...
ERROR Bootstrap failed to complete: : bootstrap process timed out: context deadline exceeded
```

2. If the output from the previous command indicates a failure, or if the bootstrap is not progressing, run the following command on node 0 and collect the output:

```
$ ssh core@<node-ip> sudo /usr/local/bin/agent-gather -O > <local_tmp_path>/agent-gather.tar.xz
```



NOTE

You only need to gather data from node 0, but gathering this data from every node can be helpful.

3. If the bootstrap completes and the cluster nodes reboot, run the following command and collect the output:

```
$ ./openshift-install --dir <install_directory> agent wait-for install-complete --log-level=debug
```

4. If the output from the previous command indicates a failure, perform the following steps:
 - a. Export the **kubeconfig** file to your environment by running the following command:

```
$ export KUBECONFIG=<install_directory>/auth/kubeconfig
```

- b. To gather information for debugging, run the following command:

```
$ oc adm must-gather
```

- c. Create a compressed file from the **must-gather** directory that was just created in your working directory by running the following command:

```
$ tar cvaf must-gather.tar.gz <must_gather_directory>
```

5. Excluding the **/auth** subdirectory, attach the installation directory used during the deployment to your support case on the [Red Hat Customer Portal](#).
6. Attach all other data gathered from this procedure to your support case.

14.3.4. Sample ZTP custom resources

Optional: You can use Zero touch provisioning (ZTP) custom resource (CR) objects to install an OpenShift Container Platform cluster with the Agent-based Installer.

You can customize the following ZTP custom resources to specify more details about your OpenShift Container Platform cluster. The following sample ZTP custom resources are for a single-node cluster.

agent-cluster-install.yaml

```
apiVersion: extensions.hive.openshift.io/v1beta1
kind: AgentClusterInstall
metadata:
  name: test-agent-cluster-install
  namespace: cluster0
spec:
  clusterDeploymentRef:
    name: ostest
  imageSetRef:
    name: openshift-4.12
  networking:
    clusterNetwork:
      - cidr: 10.128.0.0/14
        hostPrefix: 23
    serviceNetwork:
      - 172.30.0.0/16
  provisionRequirements:
    controlPlaneAgents: 1
    workerAgents: 0
    sshPublicKey: <YOUR_SSH_PUBLIC_KEY>
```

cluster-deployment.yaml

```
apiVersion: hive.openshift.io/v1
kind: ClusterDeployment
metadata:
  name: ostest
  namespace: cluster0
spec:
  baseDomain: test.metalkube.org
  clusterInstallRef:
    group: extensions.hive.openshift.io
    kind: AgentClusterInstall
    name: test-agent-cluster-install
    version: v1beta1
  clusterName: ostest
  controlPlaneConfig:
    servingCertificates: {}
  platform:
    agentBareMetal:
      agentSelector:
        matchLabels:
          bla: aaa
  pullSecretRef:
    name: pull-secret
```

cluster-image-set.yaml

```
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
```

```

metadata:
  name: openshift-4.12
spec:
  releaseImage: registry.ci.openshift.org/ocp/release:4.12.0-0.nightly-2022-06-06-025509

```

infra-env.yaml

```

apiVersion: agent-install.openshift.io/v1beta1
kind: InfraEnv
metadata:
  name: myinfraenv
  namespace: cluster0
spec:
  clusterRef:
    name: ostest
    namespace: cluster0
  pullSecretRef:
    name: pull-secret
  sshAuthorizedKey: <YOUR_SSH_PUBLIC_KEY>
  nmStateConfigLabelSelector:
    matchLabels:
      cluster0-nmstate-label-name: cluster0-nmstate-label-value

```

nmstateconfig.yaml

```

apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2
              prefix-length: 23
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.122.1
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.122.1
          next-hop-interface: eth0
          table-id: 254

```

```
interfaces:  
  - name: "eth0"  
    macAddress: 52:54:01:aa:aa:a1
```

pull-secret.yaml

```
apiVersion: v1  
kind: Secret  
type: kubernetes.io/dockerconfigjson  
metadata:  
  name: pull-secret  
  namespace: cluster0  
stringData:  
  .dockerconfigjson: 'YOUR_PULL_SECRET'
```

Additional resources

- See [Challenges of the network far edge](#) to learn more about zero touch provisioning (ZTP).

14.4. PREPARING AN AGENT-BASED INSTALLED CLUSTER FOR THE MULTICLUSTER ENGINE FOR KUBERNETES OPERATOR

You can install the multicluster engine Operator and deploy a hub cluster with the Agent-based OpenShift Container Platform Installer. The following procedure is partially automated and requires manual steps after the initial cluster is deployed.

14.4.1. Prerequisites

- You have read the following documentation:
 - [Cluster lifecycle with multicluster engine operator overview](#) .
 - [Persistent storage using local volumes](#) .
 - [Using ZTP to provision clusters at the network far edge](#) .
 - [Preparing to install with the Agent-based Installer](#) .
 - [About disconnected installation mirroring](#) .
- You have access to the internet to obtain the necessary container images.
- You have installed the OpenShift CLI (**oc**).
- If you are installing in a disconnected environment, you must have a configured local mirror registry for disconnected installation mirroring.

14.4.2. Preparing an Agent-based cluster deployment for the multicluster engine for Kubernetes Operator while disconnected

You can mirror the required OpenShift Container Platform container images, the multicluster engine Operator, and the Local Storage Operator (LSO) into your local mirror registry in a disconnected environment. Ensure that you note the local DNS hostname and port of your mirror registry.



NOTE

To mirror your OpenShift Container Platform image repository to your mirror registry, you can use either the **oc adm release image** or **oc mirror** command. In this procedure, the **oc mirror** command is used as an example.

Procedure

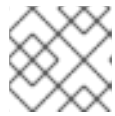
1. Create an **<assets_directory>** folder to contain valid **install-config.yaml** and **agent-config.yaml** files. This directory is used to store all the assets.
2. To mirror an OpenShift Container Platform image repository, the multicluster engine, and the LSO, create a **ImageSetConfiguration.yaml** file with the following settings:

Example ImageSetConfiguration.yaml

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4 1
storageConfig: 2
  imageURL: <your-local-registry-dns-name>:<your-local-registry-port>/mirror/oc-mirror-
  metadata 3
  skipTLS: true
mirror:
  platform:
    architectures:
      - "amd64"
    channels:
      - name: stable-4.12 4
        type: ocp
  additionalImages:
    - name: registry.redhat.io/ubi8/ubi:latest
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12 5
      packages: 6
        - name: multicluster-engine 7
        - name: local-storage-operator 8
```

- 1** Specify the maximum size, in GiB, of each file within the image set.
- 2** Set the back-end location to receive the image set metadata. This location can be a registry or local directory. You must specify **storageConfig** values unless you are using the Technology Preview OCI feature.
- 3** Set the registry URL for the storage backend.
- 4** Set the channel that contains the OpenShift Container Platform images for the version you are installing.
- 5** Set the Operator catalog that contains the OpenShift Container Platform images that you are installing.
- 6** Specify only certain Operator packages and channels to include in the image set. Remove this field to retrieve all packages in the catalog.

- 7 The multicluster engine packages and channels.
- 8 The LSO packages and channels.

**NOTE**

This file is required by the **oc mirror** command when mirroring content.

3. To mirror a specific OpenShift Container Platform image repository, the multicluster engine, and the LSO, run the following command:

```
$ oc mirror --dest-skip-tls --config ocp-mce-imageset.yaml docker://<your-local-registry-dns-name>:<your-local-registry-port>
```

4. Update the registry and certificate in the **install-config.yaml** file:

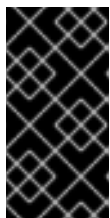
Example imageContentSources.yaml

```
imageContentSources:
- source: "quay.io/openshift-release-dev/ocp-release"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release-images"
- source: "quay.io/openshift-release-dev/ocp-v4.0-art-dev"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release"
- source: "registry.redhat.io/ubi8"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/ubi8"
- source: "registry.redhat.io/multicluster-engine"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/multicluster-engine"
- source: "registry.redhat.io/rhel8"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/rhel8"
- source: "registry.redhat.io/redhat"
  mirrors:
  - "<your-local-registry-dns-name>:<your-local-registry-port>/redhat"
```

Additionally, ensure your certificate is present in the **additionalTrustBundle** field of the **install-config.yaml**.

Example install-config.yaml

```
additionalTrustBundle: |
-----BEGIN CERTIFICATE-----
ZZZZZZZZZZZZ
-----END CERTIFICATE-----
```

**IMPORTANT**

The **oc mirror** command creates a folder called **oc-mirror-workspace** with several outputs. This includes the **imageContentSourcePolicy.yaml** file that identifies all the mirrors you need for OpenShift Container Platform and your selected Operators.

5. Generate the cluster manifests by running the following command:

```
$ openshift-install agent create cluster-manifests
```

This command updates the cluster manifests folder to include a **mirror** folder that contains your mirror configuration.

14.4.3. Preparing an Agent-based cluster deployment for the multicluster engine for Kubernetes Operator while connected

Create the required manifests for the multicluster engine Operator, the Local Storage Operator (LSO), and to deploy an agent-based OpenShift Container Platform cluster as a hub cluster.

Procedure

1. Create a sub-folder named **openshift** in the **<assets_directory>** folder. This sub-folder is used to store the extra manifests that will be applied during the installation to further customize the deployed cluster. The **<assets_directory>** folder contains all the assets including the **install-config.yaml** and **agent-config.yaml** files.

**NOTE**

The installer does not validate extra manifests.

2. For the multicluster engine, create the following manifests and save them in the **<assets_directory>/openshift** folder:

Example **mce_namespace.yaml**

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    openshift.io/cluster-monitoring: "true"
  name: multicluster-engine
```

Example **mce_operatorgroup.yaml**

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: multicluster-engine-operatorgroup
  namespace: multicluster-engine
spec:
  targetNamespaces:
    - multicluster-engine
```

Example `mce_subscription.yaml`

```

apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: multicluster-engine
  namespace: multicluster-engine
spec:
  channel: "stable-2.1"
  name: multicluster-engine
  source: redhat-operators
  sourceNamespace: openshift-marketplace

```



NOTE

You can install a distributed unit (DU) at scale with the Red Hat Advanced Cluster Management (RHACM) using the assisted installer (AI). These distributed units must be enabled in the hub cluster. The AI service requires persistent volumes (PVs), which are manually created.

- For the AI service, create the following manifests and save them in the `<assets_directory>/openshift` folder:

Example `iso_namespace.yaml`

```

apiVersion: v1
kind: Namespace
metadata:
  annotations:
    openshift.io/cluster-monitoring: "true"
  name: openshift-local-storage

```

Example `iso_operatorgroup.yaml`

```

apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: local-operator-group
  namespace: openshift-local-storage
spec:
  targetNamespaces:
    - openshift-local-storage

```

Example `iso_subscription.yaml`

```

apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: local-storage-operator
  namespace: openshift-local-storage
spec:
  installPlanApproval: Automatic

```

```
name: local-storage-operator
source: redhat-operators
sourceNamespace: openshift-marketplace
```



NOTE

After creating all the manifests, your filesystem must display as follows:

Example Filesystem

```
<assets_directory>
├── install-config.yaml
├── agent-config.yaml
├── /openshift
│   ├── mce_namespace.yaml
│   ├── mce_operatorgroup.yaml
│   ├── mce_subscription.yaml
│   ├── iso_namespace.yaml
│   ├── iso_operatorgroup.yaml
│   └── iso_subscription.yaml
```

4. Create the agent ISO image by running the following command:

```
$ openshift-install agent create image --dir <assets_directory>
```

5. When the image is ready, boot the target machine and wait for the installation to complete.
6. To monitor the installation, run the following command:

```
$ openshift-install agent wait-for install-complete --dir <assets_directory>
```



NOTE

To configure a fully functional hub cluster, you must create the following manifests and manually apply them by running the command **\$ oc apply -f <manifest-name>**. The order of the manifest creation is important and where required, the waiting condition is displayed.

7. For the PVs that are required by the AI service, create the following manifests:

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: assisted-service
  namespace: openshift-local-storage
spec:
  logLevel: Normal
  managementState: Managed
  storageClassDevices:
    - devicePaths:
      - /dev/vda
```

```

- /dev/vdb
storageClassName: assisted-service
volumeMode: Filesystem

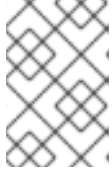
```

- Use the following command to wait for the availability of the PVs, before applying the subsequent manifests:

```

$ oc wait localvolume -n openshift-local-storage assisted-service --for condition=Available --
timeout 10m

```



NOTE

The `devicePath` is an example and may vary depending on the actual hardware configuration used.

- Create a manifest for a multicluster engine instance.

Example MultiClusterEngine.yaml

```

apiVersion: multicluster.openshift.io/v1
kind: MultiClusterEngine
metadata:
  name: multiclusterengine
spec: {}

```

- Create a manifest to enable the AI service.

Example agentserviceconfig.yaml

```

apiVersion: agent-install.openshift.io/v1beta1
kind: AgentServiceConfig
metadata:
  name: agent
  namespace: assisted-installer
spec:
  databaseStorage:
    storageClassName: assisted-service
    accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  filesystemStorage:
    storageClassName: assisted-service
    accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi

```

- Create a manifest to deploy subsequently spoke clusters.

Example clusterimageset.yaml

```

apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  name: "4.12"
spec:
  releaseImage: quay.io/openshift-release-dev/ocp-release:4.12.0-x86_64

```

12. Create a manifest to import the agent installed cluster (that hosts the multicluster engine and the Assisted Service) as the hub cluster.

Example `autoimport.yaml`

```

apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  labels:
    local-cluster: "true"
    cloud: auto-detect
    vendor: auto-detect
  name: local-cluster
spec:
  hubAcceptsClient: true

```

13. Wait for the managed cluster to be created.

```

$ oc wait -n multicluster-engine managedclusters local-cluster --for
condition=ManagedClusterJoined=True --timeout 10m

```

Verification

- To confirm that the managed cluster installation is successful, run the following command:

```

$ oc get managedcluster
NAME          HUB ACCEPTED  MANAGED CLUSTER URLS      JOINED  AVAILABLE
AGE
local-cluster true          https://<your cluster url>:6443  True   True   77m

```

Additional resources

- [The Local Storage Operator](#)

CHAPTER 15. INSTALLING ON A SINGLE NODE

15.1. PREPARING TO INSTALL ON A SINGLE NODE

15.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You have read the documentation on [selecting a cluster installation method and preparing it for users](#).

15.1.2. About OpenShift on a single node

You can create a single-node cluster with standard installation methods. OpenShift Container Platform on a single node is a specialized installation that requires the creation of a special ignition configuration ISO. The primary use case is for edge computing workloads, including intermittent connectivity, portable clouds, and 5G radio access networks (RAN) close to a base station. The major tradeoff with an installation on a single node is the lack of high availability.



IMPORTANT

The use of OpenShiftSDN with single-node OpenShift is not supported. OVN-Kubernetes is the default network plugin for single-node OpenShift deployments.

15.1.3. Requirements for installing OpenShift on a single node

Installing OpenShift Container Platform on a single node alleviates some of the requirements for high availability and large scale clusters. However, you must address the following requirements:

- **Administration host:** You must have a computer to prepare the ISO, to create the USB boot drive, and to monitor the installation.
- **Supported platforms:** Installing OpenShift Container Platform on a single node is supported on bare metal and [Certified third-party hypervisors](#). In all cases, you must specify the **platform.none: {}** parameter in the **install-config.yaml** configuration file.
- **Production-grade server:** Installing OpenShift Container Platform on a single node requires a server with sufficient resources to run OpenShift Container Platform services and a production workload.

Table 15.1. Minimum resource requirements

Profile	vCPU	Memory	Storage
Minimum	8 vCPU cores	16GB of RAM	120GB

**NOTE**

- One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio:
(threads per core × cores) × sockets = vCPUs
- Adding Operators during the installation process might increase the minimum resource requirements.

The server must have a Baseboard Management Controller (BMC) when booting with virtual media.

- Networking:** The server must have access to the internet or access to a local registry if it is not connected to a routable network. The server must have a DHCP reservation or a static IP address for the Kubernetes API, ingress route, and cluster node domain names. You must configure the DNS to resolve the IP address to each of the following fully qualified domain names (FQDN):

Table 15.2. Required DNS records

Usage	FQDN	Description
Kubernetes API	api.<cluster_name>.<base_domain>	Add a DNS A/AAAA or CNAME record. This record must be resolvable by clients external to the cluster.
Internal API	api-int.<cluster_name>.<base_domain>	Add a DNS A/AAAA or CNAME record when creating the ISO manually. This record must be resolvable by nodes within the cluster.
Ingress route	*.apps.<cluster_name>.<base_domain>	Add a wildcard DNS A/AAAA or CNAME record that targets the node. This record must be resolvable by clients external to the cluster.

Without persistent IP addresses, communications between the **apiserver** and **etcd** might fail.

15.2. INSTALLING OPENSIFT ON A SINGLE NODE

You can install single-node OpenShift using the web-based Assisted Installer and a discovery ISO that you generate using the Assisted Installer. You can also install single-node OpenShift by using **coreos-installer** to generate the installation ISO.

15.2.1. Installing single-node OpenShift using the Assisted Installer

To install OpenShift Container Platform on a single node, use the web-based Assisted Installer wizard to guide you through the process and manage the installation.

15.2.1.1. Generating the discovery ISO with the Assisted Installer

Installing OpenShift Container Platform on a single node requires a discovery ISO, which the Assisted Installer can generate.

Procedure

1. On the administration host, open a browser and navigate to [Red Hat OpenShift Cluster Manager](#).
2. Click **Create Cluster** to create a new cluster.
3. In the **Cluster name** field, enter a name for the cluster.
4. In the **Base domain** field, enter a base domain. For example:

```
example.com
```

All DNS records must be subdomains of this base domain and include the cluster name, for example:

```
<cluster-name>.example.com
```



NOTE

You cannot change the base domain or cluster name after cluster installation.

5. Select **Install single node OpenShift (SNO)** and complete the rest of the wizard steps. Download the discovery ISO.
6. Make a note of the discovery ISO URL for installing with virtual media.



NOTE

If you enable OpenShift Virtualization during this process, you must have a second local storage device of at least 50GiB for your virtual machines.

Additional resources

- [Persistent storage using logical volume manager storage](#)
- [What you can do with OpenShift Virtualization](#)

15.2.1.2. Installing single-node OpenShift with the Assisted Installer

Use the Assisted Installer to install the single-node cluster.

Procedure

1. Attach the RHCOS discovery ISO to the target host.
2. Configure the boot drive order in the server BIOS settings to boot from the attached discovery ISO and then reboot the server.

3. On the administration host, return to the browser. Wait for the host to appear in the list of discovered hosts. If necessary, reload the [Assisted Clusters](#) page and select the cluster name.
4. Complete the install wizard steps. Add networking details, including a subnet from the available subnets. Add the SSH public key if necessary.
5. Monitor the installation's progress. Watch the cluster events. After the installation process finishes writing the operating system image to the server's hard disk, the server restarts.
6. Remove the discovery ISO, and reset the server to boot from the installation drive. The server restarts several times automatically, deploying the control plane.

Additional resources

- [Creating a bootable ISO image on a USB drive](#)
- [Booting from an HTTP-hosted ISO image using the Redfish API](#)
- [Adding worker nodes to single-node OpenShift clusters](#)

15.2.2. Installing single-node OpenShift manually

To install OpenShift Container Platform on a single node, first generate the installation ISO, and then boot the server from the ISO. You can monitor the installation using the **openshift-install** installation program.

15.2.2.1. Generating the installation ISO with coreos-installer

Installing OpenShift Container Platform on a single node requires an installation ISO, which you can generate with the following procedure.

Prerequisites

- Install **podman**.

Procedure

1. Set the OpenShift Container Platform version:

```
$ OCP_VERSION=<ocp_version> 1
```

- 1 Replace **<ocp_version>** with the current version, for example, **latest-4.12**

2. Set the host architecture:

```
$ ARCH=<architecture> 1
```

- 1 Replace **<architecture>** with the target host architecture, for example, **aarch64** or **x86_64**.

3. Download the OpenShift Container Platform client (**oc**) and make it available for use by entering the following commands:

■

```
$ curl -k https://mirror.openshift.com/pub/openshift-  
v4/clients/ocp/$OCP_VERSION/openshift-client-linux.tar.gz -o oc.tar.gz
```

```
$ tar xzf oc.tar.gz
```

```
$ chmod +x oc
```

- Download the OpenShift Container Platform installer and make it available for use by entering the following commands:

```
$ curl -k https://mirror.openshift.com/pub/openshift-  
v4/clients/ocp/$OCP_VERSION/openshift-install-linux.tar.gz -o openshift-install-linux.tar.gz
```

```
$ tar zxvf openshift-install-linux.tar.gz
```

```
$ chmod +x openshift-install
```

- Retrieve the RHCOS ISO URL by running the following command:

```
$ ISO_URL=$(./openshift-install coreos print-stream-json | grep location | grep $ARCH | grep  
iso | cut -d\" -f4)
```

- Download the RHCOS ISO:

```
$ curl -L $ISO_URL -o rhcos-live.iso
```

- Prepare the **install-config.yaml** file:

```
apiVersion: v1  
baseDomain: <domain> ①  
compute:  
- architecture: amd64 ②  
  name: worker  
  replicas: 0 ③  
controlPlane:  
  architecture: amd64  
  name: master  
  replicas: 1 ④  
metadata:  
  name: <name> ⑤  
networking: ⑥  
  clusterNetwork:  
  - cidr: 10.128.0.0/14  
    hostPrefix: 23  
  machineNetwork:  
  - cidr: 10.0.0.0/16 ⑦  
  networkType: OVNKubernetes  
serviceNetwork:  
- 172.30.0.0/16  
platform:  
  none: {}
```

```
bootstrapInPlace:
  installationDisk: /dev/disk/by-id/<disk_id> 8
  pullSecret: '<pull_secret>' 9
  sshKey: |
    <ssh_key> 10
```

- 1 Add the cluster domain name.
- 2 Set the architecture to **arm64** for 64-bit ARM or **amd64** for 64-bit x86 architectures. This needs to be set explicitly to the target host architecture.
- 3 Set the **compute** replicas to **0**. This makes the control plane node schedulable.
- 4 Set the **controlPlane** replicas to **1**. In conjunction with the previous **compute** setting, this setting ensures the cluster runs on a single node.
- 5 Set the **metadata** name to the cluster name.
- 6 Set the **networking** details. OVN-Kubernetes is the only allowed network plugin type for single-node clusters.
- 7 Set the **cidr** value to match the subnet of the single-node OpenShift cluster.
- 8 Set the path to the installation disk drive, for example, **/dev/disk/by-id/wwn-0x64cd98f04fde100024684cf3034da5c2**.
- 9 Copy the [pull secret from the Red Hat OpenShift Cluster Manager](#) and add the contents to this configuration setting.
- 10 Add the public SSH key from the administration host so that you can log in to the cluster after installation.

8. Generate OpenShift Container Platform assets by running the following commands:

```
$ mkdir ocp
```

```
$ cp install-config.yaml ocp
```

```
$ ./openshift-install --dir=ocp create single-node-ignition-config
```

9. Embed the ignition data into the RHCOS ISO by running the following commands:

```
$ alias coreos-installer='podman run --privileged --pull always --rm \
  -v /dev:/dev -v /run/udev:/run/udev -v $PWD:/data \
  -w /data quay.io/coreos/coreos-installer:release'
```

```
$ coreos-installer iso ignition embed -fi ocp/bootstrap-in-place-for-live-iso.ign rhcos-live.iso
```

Additional resources

- See [Enabling cluster capabilities](#) for more information about enabling cluster capabilities that were disabled prior to installation.

- See [Optional cluster capabilities in OpenShift Container Platform OpenShift Container Platform 4.12](#) for more information about the features provided by each capability.

15.2.2.2. Monitoring the cluster installation using openshift-install

Use **openshift-install** to monitor the progress of the single-node cluster installation.

Procedure

1. Attach the modified RHCOS installation ISO to the target host.
2. Configure the boot drive order in the server BIOS settings to boot from the attached discovery ISO and then reboot the server.
3. On the administration host, monitor the installation by running the following command:

```
$ ./openshift-install --dir=ocp wait-for install-complete
```

The server restarts several times while deploying the control plane.

Verification

- After the installation is complete, check the environment by running the following command:

```
$ export KUBECONFIG=ocp/auth/kubeconfig
```

```
$ oc get nodes
```

Example output

```
NAME                STATUS ROLES    AGE  VERSION
control-plane.example.com Ready  master,worker 10m  v1.25.0
```

Additional resources

- [Creating a bootable ISO image on a USB drive](#)
- [Booting from an HTTP-hosted ISO image using the Redfish API](#)
- [Adding worker nodes to single-node OpenShift clusters](#)

15.2.3. Creating a bootable ISO image on a USB drive

You can install software using a bootable USB drive that contains an ISO image. Booting the server with the USB drive prepares the server for the software installation.

Procedure

1. On the administration host, insert a USB drive into a USB port.
2. Create a bootable USB drive, for example:

```
# dd if=<path_to_iso> of=<path_to_usb> status=progress
```

where:

<path_to_iso>

is the relative path to the downloaded ISO file, for example, **rhcos-live.iso**.

<path_to_usb>

is the location of the connected USB drive, for example, **/dev/sdb**.

After the ISO is copied to the USB drive, you can use the USB drive to install software on the server.

15.2.4. Booting from an HTTP-hosted ISO image using the Redfish API

You can provision hosts in your network using ISOs that you install using the Redfish Baseboard Management Controller (BMC) API.

Prerequisites

1. Download the installation Red Hat Enterprise Linux CoreOS (RHCOS) ISO.

Procedure

1. Copy the ISO file to an HTTP server accessible in your network.
2. Boot the host from the hosted ISO file, for example:
 - a. Call the redfish API to set the hosted ISO as the **VirtualMedia** boot media by running the following command:

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"Image": "<hosted_iso_file>",
"Inserted": true}' -H "Content-Type: application/json" -X POST
<host_bmc_address>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia/CD/Actions/Vi
rtualMedia.InsertMedia
```

Where:

<bmc_username>:<bmc_password>

Is the username and password for the target host BMC.

<hosted_iso_file>

Is the URL for the hosted installation ISO, for example:

<http://webserver.example.com/rhcos-live-minimal.iso>. The ISO must be accessible from the target host machine.

<host_bmc_address>

Is the BMC IP address of the target host machine.

- b. Set the host to boot from the **VirtualMedia** device by running the following command:

```
$ curl -k -u <bmc_username>:<bmc_password> -X PATCH -H 'Content-Type:
application/json' -d '{"Boot": {"BootSourceOverrideTarget": "Cd",
"BootSourceOverrideMode": "UEFI", "BootSourceOverrideEnabled": "Once"}}'
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1
```

- c. Reboot the host:

-

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"ResetType": "ForceRestart"}' -H
'Content-type: application/json' -X POST
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1/Actions/ComputerSystem.Reset
```

- d. Optional: If the host is powered off, you can boot it using the **{"ResetType": "On"}** switch. Run the following command:

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"ResetType": "On"}' -H 'Content-
type: application/json' -X POST
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1/Actions/ComputerSystem.Reset
```

15.2.5. Creating a custom live RHCOS ISO for remote server access

In some cases, you cannot attach an external disk drive to a server, however, you need to access the server remotely to provision a node. It is recommended to enable SSH access to the server. You can create a live RHCOS ISO with SSHd enabled and with predefined credentials so that you can access the server after it boots.

Prerequisites

- You installed the **butane** utility.

Procedure

- Download the **coreos-installer** binary from the **coreos-installer** image [mirror](#) page.
- Download the latest live RHCOS ISO from [mirror.openshift.com](#).
- Create the **embedded.yaml** file that the **butane** utility uses to create the Ignition file:

```
variant: openshift
version: 4.12.0
metadata:
  name: sshd
  labels:
    machineconfiguration.openshift.io/role: worker
passwd:
  users:
    - name: core 1
      ssh_authorized_keys:
        - '<ssh_key>'
```

- 1** The **core** user has sudo privileges.

- Run the **butane** utility to create the Ignition file using the following command:

```
$ butane -pr embedded.yaml -o embedded.ign
```

- After the Ignition file is created, you can include the configuration in a new live RHCOS ISO, which is named **rhcos-sshd-4.12.0-x86_64-live.x86_64.iso**, with the **coreos-installer** utility:


```
$ coreos-installer iso ignition embed -i embedded.ign rhcos-4.12.0-x86_64-live.x86_64.iso -o
rhcos-sshd-4.12.0-x86_64-live.x86_64.iso
```

Verification

- Check that the custom live ISO can be used to boot the server by running the following command:

```
# coreos-installer iso ignition show rhcos-sshd-4.12.0-x86_64-live.x86_64.iso
```

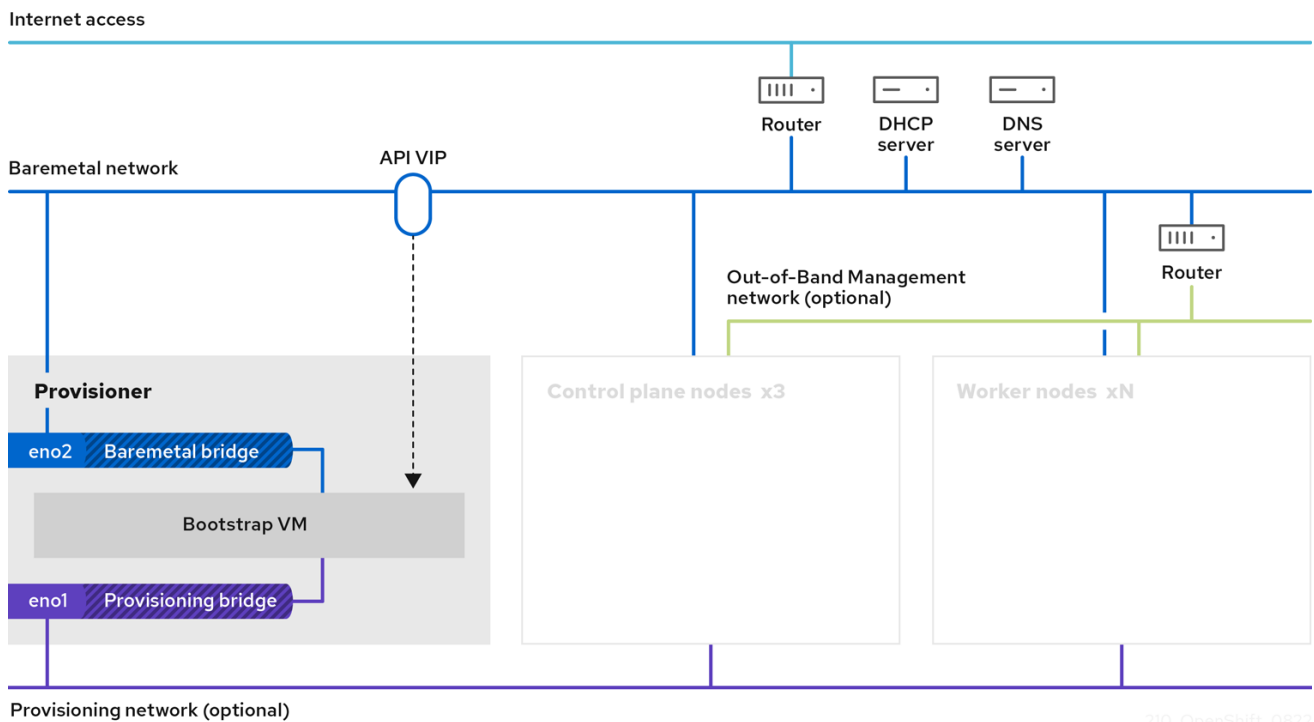
Example output

```
{
  "ignition": {
    "version": "3.2.0"
  },
  "passwd": {
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [
          "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQACZnG8AlzIDAhyENpK2qKiTT8EbRWOrz7NXjRzo
pbPu215mocaJgjjwJjh1cYhgPhpAp6M/ttTk714OI7g4588Apx4bwJep6oWTU35LkY8ZxkGVPJL
8kVITdKQviDv3XX12l4QfnDom4tm4gVbRH0gNT1wzhnLP+LKym2Ohr9D7p9NBnAdro6k++X
WgkDeijLRUTwdEyWunldW1f8G0Mg8Y1Xzr13BUo3+8aey7HLKJMDtobkz/C8ESYA/f7HJc5Fx
F0XbapWWovSSDJrr9OmIL9f4TfE+cQk3s+eoKiz2bgNPRgEEwihVbGsCN4grA+RzLCAOpec+
2dTJrQvFqsD alosadag@sonnelicht.local"
        ]
      }
    ]
  }
}
```

CHAPTER 16. DEPLOYING INSTALLER-PROVISIONED CLUSTERS ON BARE METAL

16.1. OVERVIEW

Installer-provisioned installation on bare metal nodes deploys and configures the infrastructure that an OpenShift Container Platform cluster runs on. This guide provides a methodology to achieving a successful installer-provisioned bare-metal installation. The following diagram illustrates the installation environment in phase 1 of deployment:



For the installation, the key elements in the previous diagram are:

- **Provisioner:** A physical machine that runs the installation program and hosts the bootstrap VM that deploys the control plane of a new OpenShift Container Platform cluster.
- **Bootstrap VM:** A virtual machine used in the process of deploying an OpenShift Container Platform cluster.
- **Network bridges:** The bootstrap VM connects to the bare metal network and to the provisioning network, if present, via network bridges, **eno1** and **eno2**.
- **API VIP:** An API virtual IP address (VIP) is used to provide failover of the API server across the control plane nodes. The API VIP first resides on the bootstrap VM. A script generates the **keepalived.conf** configuration file before launching the service. The VIP moves to one of the control plane nodes after the bootstrap process has completed and the bootstrap VM stops.

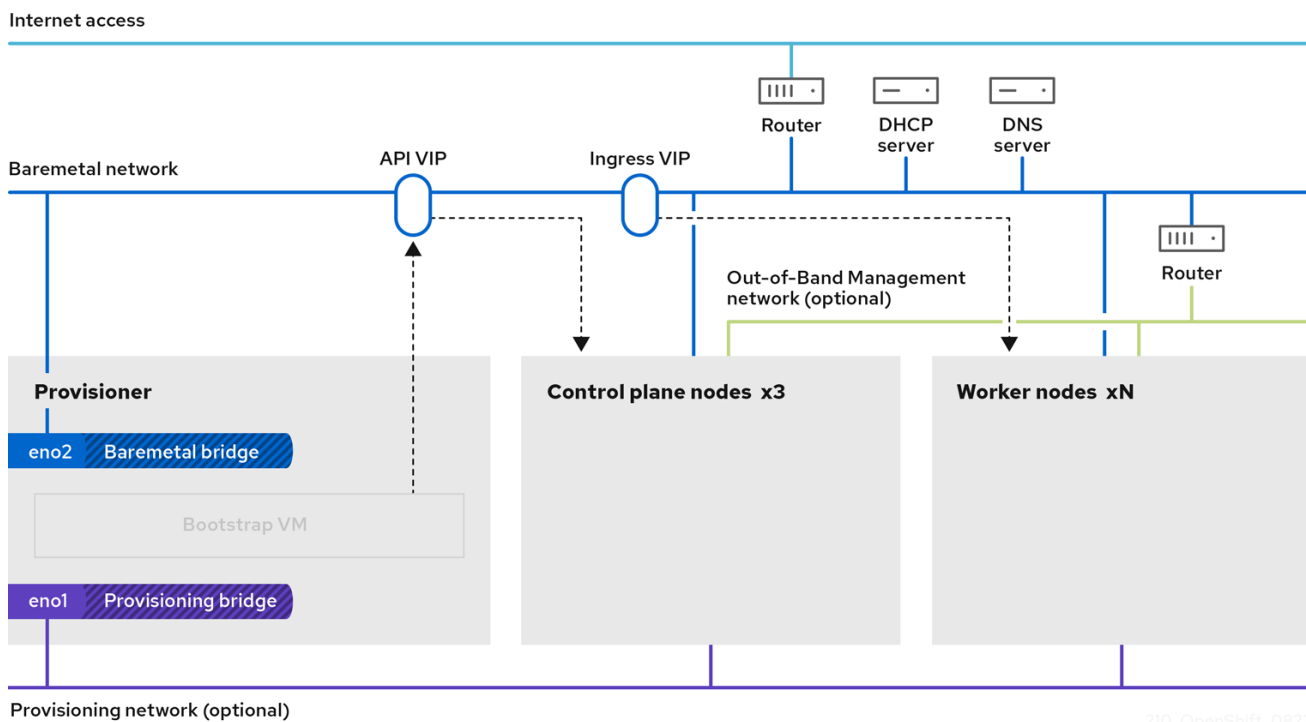
In phase 2 of the deployment, the provisioner destroys the bootstrap VM automatically and moves the virtual IP addresses (VIPs) to the appropriate nodes.

The **keepalived.conf** file sets the control plane machines with a lower Virtual Router Redundancy Protocol (VRRP) priority than the bootstrap VM, which ensures that the API on the control plane machines is fully functional before the API VIP moves from the bootstrap VM to the control plane. Once

the API VIP moves to one of the control plane nodes, traffic sent from external clients to the API VIP routes to an **haproxy** load balancer running on that control plane node. This instance of **haproxy** load balances the API VIP traffic across the control plane nodes.

The Ingress VIP moves to the worker nodes. The **keepalived** instance also manages the Ingress VIP.

The following diagram illustrates phase 2 of deployment:



210_OpenShift_0822

After this point, the node used by the provisioner can be removed or repurposed. From here, all additional provisioning tasks are carried out by the control plane.



IMPORTANT

The provisioning network is optional, but it is required for PXE booting. If you deploy without a provisioning network, you must use a virtual media baseboard management controller (BMC) addressing option such as **redfish-virtualmedia** or **idrac-virtualmedia**.

16.2. PREREQUISITES

Installer-provisioned installation of OpenShift Container Platform requires:

1. One provisioner node with Red Hat Enterprise Linux (RHEL) 8.x installed. The provisioner can be removed after installation.
2. Three control plane nodes
3. Baseboard management controller (BMC) access to each node
4. At least one network:
 - a. One required routable network
 - b. One optional provisioning network

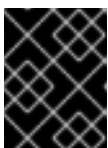
- c. One optional management network

Before starting an installer-provisioned installation of OpenShift Container Platform, ensure the hardware environment meets the following requirements.

16.2.1. Node requirements

Installer-provisioned installation involves a number of hardware node requirements:

- **CPU architecture:** All nodes must use **x86_64** or **aarch64** CPU architecture.
- **Similar nodes:** Red Hat recommends nodes have an identical configuration per role. That is, Red Hat recommends nodes be the same brand and model with the same CPU, memory, and storage configuration.
- **Baseboard Management Controller:** The **provisioner** node must be able to access the baseboard management controller (BMC) of each OpenShift Container Platform cluster node. You may use IPMI, Redfish, or a proprietary protocol.
- **Latest generation:** Nodes must be of the most recent generation. Installer-provisioned installation relies on BMC protocols, which must be compatible across nodes. Additionally, RHEL 8 ships with the most recent drivers for RAID controllers. Ensure that the nodes are recent enough to support RHEL 8 for the **provisioner** node and RHCOS 8 for the control plane and worker nodes.
- **Registry node:** (Optional) If setting up a disconnected mirrored registry, it is recommended the registry reside in its own node.
- **Provisioner node:** Installer-provisioned installation requires one **provisioner** node.
- **Control plane:** Installer-provisioned installation requires three control plane nodes for high availability. You can deploy an OpenShift Container Platform cluster with only three control plane nodes, making the control plane nodes schedulable as worker nodes. Smaller clusters are more resource efficient for administrators and developers during development, production, and testing.
- **Worker nodes:** While not required, a typical production cluster has two or more worker nodes.



IMPORTANT

Do not deploy a cluster with only one worker node, because the cluster will deploy with routers and ingress traffic in a degraded state.

- **Network interfaces:** Each node must have at least one network interface for the routable **baremetal** network. Each node must have one network interface for a **provisioning** network when using the **provisioning** network for deployment. Using the **provisioning** network is the default configuration.
- **Unified Extensible Firmware Interface (UEFI):** Installer-provisioned installation requires UEFI boot on all OpenShift Container Platform nodes when using IPv6 addressing on the **provisioning** network. In addition, UEFI Device PXE Settings must be set to use the IPv6 protocol on the **provisioning** network NIC, but omitting the **provisioning** network removes this requirement.



IMPORTANT

When starting the installation from virtual media such as an ISO image, delete all old UEFI boot table entries. If the boot table includes entries that are not generic entries provided by the firmware, the installation might fail.

- **Secure Boot:** Many production scenarios require nodes with Secure Boot enabled to verify the node only boots with trusted software, such as UEFI firmware drivers, EFI applications, and the operating system. You may deploy with Secure Boot manually or managed.
 1. **Manually:** To deploy an OpenShift Container Platform cluster with Secure Boot manually, you must enable UEFI boot mode and Secure Boot on each control plane node and each worker node. Red Hat supports Secure Boot with manually enabled UEFI and Secure Boot only when installer-provisioned installations use Redfish virtual media. See "Configuring nodes for Secure Boot manually" in the "Configuring nodes" section for additional details.
 2. **Managed:** To deploy an OpenShift Container Platform cluster with managed Secure Boot, you must set the **bootMode** value to **UEFISecureBoot** in the **install-config.yaml** file. Red Hat only supports installer-provisioned installation with managed Secure Boot on 10th generation HPE hardware and 13th generation Dell hardware running firmware version **2.75.75.75** or greater. Deploying with managed Secure Boot does not require Redfish virtual media. See "Configuring managed Secure Boot" in the "Setting up the environment for an OpenShift installation" section for details.



NOTE

Red Hat does not support Secure Boot with self-generated keys.

16.2.2. Planning a bare metal cluster for OpenShift Virtualization

If you will use OpenShift Virtualization, it is important to be aware of several requirements before you install your bare metal cluster.

- If you want to use live migration features, you must have multiple worker nodes *at the time of cluster installation*. This is because live migration requires the cluster-level high availability (HA) flag to be set to true. The HA flag is set when a cluster is installed and cannot be changed afterwards. If there are fewer than two worker nodes defined when you install your cluster, the HA flag is set to false for the life of the cluster.



NOTE

You can install OpenShift Virtualization on a single-node cluster, but single-node OpenShift does not support high availability.

- Live migration requires shared storage. Storage for OpenShift Virtualization must support and use the ReadWriteMany (RWX) access mode.
- If you plan to use Single Root I/O Virtualization (SR-IOV), ensure that your network interface controllers (NICs) are supported by OpenShift Container Platform.

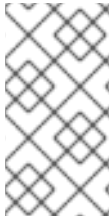
Additional resources

- [Preparing your cluster for OpenShift Virtualization](#)
- [About Single Root I/O Virtualization \(SR-IOV\) hardware networks](#)

- [Connecting a virtual machine to an SR-IOV network](#)

16.2.3. Firmware requirements for installing with virtual media

The installation program for installer-provisioned OpenShift Container Platform clusters validates the hardware and firmware compatibility with Redfish virtual media. The installation program does not begin installation on a node if the node firmware is not compatible. The following tables list the minimum firmware versions tested and verified to work for installer-provisioned OpenShift Container Platform clusters deployed by using Redfish virtual media.



NOTE

Red Hat does not test every combination of firmware, hardware, or other third-party components. For further information about third-party support, see [Red Hat third-party support policy](#). For information about updating the firmware, see the hardware documentation for the nodes or contact the hardware vendor.

Table 16.1. Firmware compatibility for HP hardware with Redfish virtual media

Model	Management	Firmware versions
10th Generation	iLO5	2.63 or later

Table 16.2. Firmware compatibility for Dell hardware with Redfish virtual media

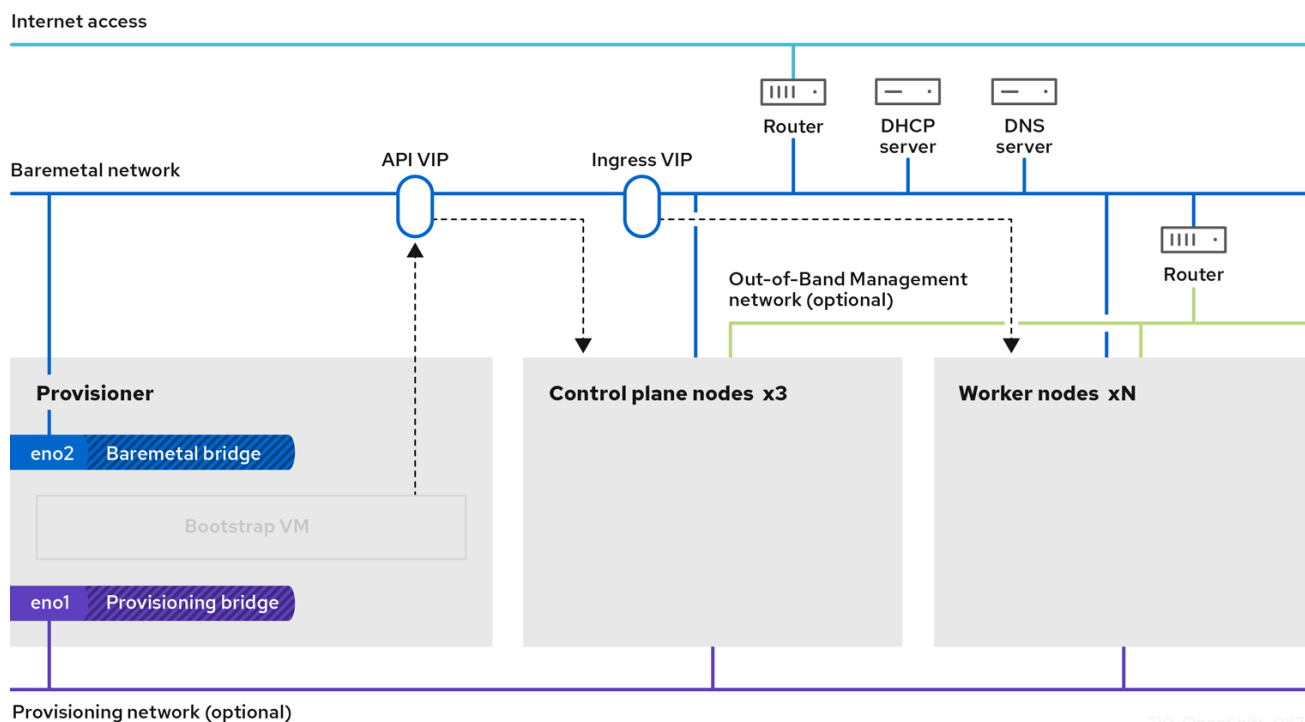
Model	Management	Firmware versions
15th Generation	iDRAC 9	v6.10.30.00
14th Generation	iDRAC 9	v6.10.30.00
13th Generation	iDRAC 8	v2.75.75.75 or later

Additional resources

[Unable to discover new bare metal hosts using the BMC](#)

16.2.4. Network requirements

Installer-provisioned installation of OpenShift Container Platform involves several network requirements. First, installer-provisioned installation involves an optional non-routable **provisioning** network for provisioning the operating system on each bare metal node. Second, installer-provisioned installation involves a routable **baremetal** network.



16.2.4.1. Increase the network MTU

Before deploying OpenShift Container Platform, increase the network maximum transmission unit (MTU) to 1500 or more. If the MTU is lower than 1500, the Ironic image that is used to boot the node might fail to communicate with the Ironic inspector pod, and inspection will fail. If this occurs, installation stops because the nodes are not available for installation.

16.2.4.2. Configuring NICs

OpenShift Container Platform deploys with two networks:

- **provisioning:** The **provisioning** network is an optional non-routable network used for provisioning the underlying operating system on each node that is a part of the OpenShift Container Platform cluster. The network interface for the **provisioning** network on each cluster node must have the BIOS or UEFI configured to PXE boot.

The **provisioningNetworkInterface** configuration setting specifies the **provisioning** network NIC name on the control plane nodes, which must be identical on the control plane nodes. The **bootMACAddress** configuration setting provides a means to specify a particular NIC on each node for the **provisioning** network.

The **provisioning** network is optional, but it is required for PXE booting. If you deploy without a **provisioning** network, you must use a virtual media BMC addressing option such as **redfish-virtualmedia** or **idrac-virtualmedia**.

- **baremetal:** The **baremetal** network is a routable network. You can use any NIC to interface with the **baremetal** network provided the NIC is not configured to use the **provisioning** network.



IMPORTANT

When using a VLAN, each NIC must be on a separate VLAN corresponding to the appropriate network.

16.2.4.3. DNS requirements

Clients access the OpenShift Container Platform cluster nodes over the **baremetal** network. A network administrator must configure a subdomain or subzone where the canonical name extension is the cluster name.

```
<cluster_name>.<base_domain>
```

For example:

```
test-cluster.example.com
```

OpenShift Container Platform includes functionality that uses cluster membership information to generate A/AAAA records. This resolves the node names to their IP addresses. After the nodes are registered with the API, the cluster can disperse node information without using CoreDNS-mDNS. This eliminates the network traffic associated with multicast DNS.

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard ingress API

A/AAAA records are used for name resolution and PTR records are used for reverse name resolution. Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records or DHCP to set the hostnames for all the nodes.

Installer-provisioned installation includes functionality that uses cluster membership information to generate A/AAAA records. This resolves the node names to their IP addresses. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 16.3. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	An A/AAAA record and a PTR record identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Routes	*.apps.<cluster_name>.<base_domain>.	The wildcard A/AAAA record refers to the application ingress load balancer. The application ingress load balancer targets the nodes that run the Ingress Controller pods. The Ingress Controller pods run on the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster. For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.

TIP

You can use the **dig** command to verify DNS resolution.

16.2.4.4. Dynamic Host Configuration Protocol (DHCP) requirements

By default, installer-provisioned installation deploys **ironic-dnsmasq** with DHCP enabled for the **provisioning** network. No other DHCP servers should be running on the **provisioning** network when the **provisioningNetwork** configuration setting is set to **managed**, which is the default value. If you have a DHCP server running on the **provisioning** network, you must set the **provisioningNetwork** configuration setting to **unmanaged** in the **install-config.yaml** file.

Network administrators must reserve IP addresses for each node in the OpenShift Container Platform cluster for the **baremetal** network on an external DHCP server.

16.2.4.5. Reserving IP addresses for nodes with the DHCP server

For the **baremetal** network, a network administrator must reserve a number of IP addresses, including:

1. Two unique virtual IP addresses.
 - One virtual IP address for the API endpoint.
 - One virtual IP address for the wildcard ingress endpoint.
2. One IP address for the provisioner node.
3. One IP address for each control plane node.
4. One IP address for each worker node, if applicable.

**RESERVING IP ADDRESSES SO THEY BECOME STATIC IP ADDRESSES**

Some administrators prefer to use static IP addresses so that each node's IP address remains constant in the absence of a DHCP server. To configure static IP addresses with NMState, see "(Optional) Configuring host network interfaces" in the "Setting up the environment for an OpenShift installation" section.

**NETWORKING BETWEEN EXTERNAL LOAD BALANCERS AND CONTROL PLANE NODES**

External load balancing services and the control plane nodes must run on the same L2 network, and on the same VLAN when using VLANs to route traffic between the load balancing services and the control plane nodes.

**IMPORTANT**

The storage interface requires a DHCP reservation or a static IP.

The following table provides an exemplary embodiment of fully qualified domain names. The API and Nameserver addresses begin with canonical name extensions. The hostnames of the control plane and worker nodes are exemplary, so you can use any host naming convention you prefer.

Usage	Host Name	IP
API	api.<cluster_name>.<base_domain>	<ip>
Ingress LB (apps)	*.apps.<cluster_name>.<base_domain>	<ip>
Provisioner node	provisioner.<cluster_name>.<base_domain>	<ip>
Control-plane-0	openshift-control-plane-0.<cluster_name>.<base_domain>	<ip>
Control-plane-1	openshift-control-plane-1.<cluster_name>.<base_domain>	<ip>
Control-plane-2	openshift-control-plane-2.<cluster_name>.<base_domain>	<ip>
Worker-0	openshift-worker-0.<cluster_name>.<base_domain>	<ip>
Worker-1	openshift-worker-1.<cluster_name>.<base_domain>	<ip>
Worker-n	openshift-worker-n.<cluster_name>.<base_domain>	<ip>



NOTE

If you do not create DHCP reservations, the installer requires reverse DNS resolution to set the hostnames for the Kubernetes API node, the provisioner node, the control plane nodes, and the worker nodes.

16.2.4.6. Provisioner node requirements

You must specify the MAC address for the provisioner node in your installation configuration. The **bootMacAddress** specification is typically associated with PXE network booting. However, the Ironic provisioning service also requires the **bootMacAddress** specification to identify nodes during the inspection of the cluster, or during node redeployment in the cluster.

The provisioner node requires layer 2 connectivity for network booting, DHCP and DNS resolution, and local network communication. The provisioner node requires layer 3 connectivity for virtual media booting.

16.2.4.7. Network Time Protocol (NTP)

Each OpenShift Container Platform node in the cluster must have access to an NTP server. OpenShift Container Platform nodes use NTP to synchronize their clocks. For example, cluster nodes use SSL certificates that require validation, which might fail if the date and time between the nodes are not in sync.



IMPORTANT

Define a consistent clock date and time format in each cluster node's BIOS settings, or installation might fail.

You can reconfigure the control plane nodes to act as NTP servers on disconnected clusters, and reconfigure worker nodes to retrieve time from the control plane nodes.

16.2.4.8. Port access for the out-of-band management IP address

The out-of-band management IP address is on a separate network from the node. To ensure that the out-of-band management can communicate with the provisioner during installation, the out-of-band management IP address must be granted access to port **80** on the bootstrap host and port **6180** on the OpenShift Container Platform control plane hosts. TLS port **6183** is required for virtual media installation, for example, via Redfish.

16.2.5. Configuring nodes

Configuring nodes when using the provisioning network

Each node in the cluster requires the following configuration for proper installation.



WARNING

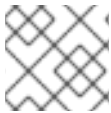
A mismatch between nodes will cause an installation failure.

While the cluster nodes can contain more than two NICs, the installation process only focuses on the first two NICs. In the following table, NIC1 is a non-routable network (**provisioning**) that is only used for the installation of the OpenShift Container Platform cluster.

NIC	Network	VLAN
NIC1	provisioning	<provisioning_vlan>
NIC2	baremetal	<baremetal_vlan>

The Red Hat Enterprise Linux (RHEL) 8.x installation process on the provisioner node might vary. To install Red Hat Enterprise Linux (RHEL) 8.x using a local Satellite server or a PXE server, PXE-enable NIC2.

PXE	Boot order
NIC1 PXE-enabled provisioning network	1
NIC2 baremetal network. PXE-enabled is optional.	2

**NOTE**

Ensure PXE is disabled on all other NICs.

Configure the control plane and worker nodes as follows:

PXE	Boot order
NIC1 PXE-enabled (provisioning network)	1

Configuring nodes without the provisioning network

The installation process requires one NIC:

NIC	Network	VLAN
NICx	baremetal	<baremetal_vlan>

NICx is a routable network (**baremetal**) that is used for the installation of the OpenShift Container Platform cluster, and routable to the internet.

**IMPORTANT**

The **provisioning** network is optional, but it is required for PXE booting. If you deploy without a **provisioning** network, you must use a virtual media BMC addressing option such as **redfish-virtualmedia** or **idrac-virtualmedia**.

Configuring nodes for Secure Boot manually

Secure Boot prevents a node from booting unless it verifies the node is using only trusted software, such as UEFI firmware drivers, EFI applications, and the operating system.

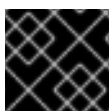
**NOTE**

Red Hat only supports manually configured Secure Boot when deploying with Redfish virtual media.

To enable Secure Boot manually, refer to the hardware guide for the node and execute the following:

Procedure

1. Boot the node and enter the BIOS menu.
2. Set the node's boot mode to **UEFI Enabled**.
3. Enable Secure Boot.

**IMPORTANT**

Red Hat does not support Secure Boot with self-generated keys.

16.2.6. Out-of-band management

Nodes typically have an additional NIC used by the baseboard management controllers (BMCs). These BMCs must be accessible from the provisioner node.

Each node must be accessible via out-of-band management. When using an out-of-band management network, the provisioner node requires access to the out-of-band management network for a successful OpenShift Container Platform installation.

The out-of-band management setup is out of scope for this document. Using a separate management network for out-of-band management can enhance performance and improve security. However, using the provisioning network or the bare metal network are valid options.



NOTE

The bootstrap VM features a maximum of two network interfaces. If you configure a separate management network for out-of-band management, and you are using a provisioning network, the bootstrap VM requires routing access to the management network through one of the network interfaces. In this scenario, the bootstrap VM can then access three networks:

- the bare metal network
- the provisioning network
- the management network routed through one of the network interfaces

16.2.7. Required data for installation

Prior to the installation of the OpenShift Container Platform cluster, gather the following information from all cluster nodes:

- Out-of-band management IP
 - Examples
 - Dell (iDRAC) IP
 - HP (iLO) IP
 - Fujitsu (iRMC) IP

When using the provisioning network

- NIC (**provisioning**) MAC address
- NIC (**baremetal**) MAC address

When omitting the provisioning network

- NIC (**baremetal**) MAC address

16.2.8. Validation checklist for nodes

When using the provisioning network

- NIC1 VLAN is configured for the **provisioning** network.
- NIC1 for the **provisioning** network is PXE-enabled on the provisioner, control plane, and worker nodes.
- NIC2 VLAN is configured for the **baremetal** network.
- PXE has been disabled on all other NICs.
- DNS is configured with API and Ingress endpoints.
- Control plane and worker nodes are configured.
- All nodes accessible via out-of-band management.
- (Optional) A separate management network has been created.
- Required data for installation.

When omitting the provisioning network

- NIC1 VLAN is configured for the **baremetal** network.
- DNS is configured with API and Ingress endpoints.
- Control plane and worker nodes are configured.
- All nodes accessible via out-of-band management.
- (Optional) A separate management network has been created.
- Required data for installation.

16.3. SETTING UP THE ENVIRONMENT FOR AN OPENSIFT INSTALLATION

16.3.1. Installing RHEL on the provisioner node

With the configuration of the prerequisites complete, the next step is to install RHEL 8.x on the provisioner node. The installer uses the provisioner node as the orchestrator while installing the OpenShift Container Platform cluster. For the purposes of this document, installing RHEL on the provisioner node is out of scope. However, options include but are not limited to using a RHEL Satellite server, PXE, or installation media.

16.3.2. Preparing the provisioner node for OpenShift Container Platform installation

Perform the following steps to prepare the environment.

Procedure

1. Log in to the provisioner node via **ssh**.
2. Create a non-root user (**kni**) and provide that user with **sudo** privileges:

```
# useradd kni
```

-

```
# passwd kni
```

```
# echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
```

```
# chmod 0440 /etc/sudoers.d/kni
```

3. Create an **ssh** key for the new user:

```
# su - kni -c "ssh-keygen -t ed25519 -f /home/kni/.ssh/id_rsa -N ""
```

4. Log in as the new user on the provisioner node:

```
# su - kni
```

5. Use Red Hat Subscription Manager to register the provisioner node:

```
$ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
$ sudo subscription-manager repos --enable=rhel-8-for-<architecture>-appstream-rpms --
enable=rhel-8-for-<architecture>-baseos-rpms
```



NOTE

For more information about Red Hat Subscription Manager, see [Using and Configuring Red Hat Subscription Manager](#).

6. Install the following packages:

```
$ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
```

7. Modify the user to add the **libvirt** group to the newly created user:

```
$ sudo usermod --append --groups libvirt <user>
```

8. Restart **firewalld** and enable the **http** service:

```
$ sudo systemctl start firewalld
```

```
$ sudo firewall-cmd --zone=public --add-service=http --permanent
```

```
$ sudo firewall-cmd --reload
```

9. Start and enable the **libvirtd** service:

```
$ sudo systemctl enable libvirtd --now
```

10. Create the **default** storage pool and start it:

```
$ sudo virsh pool-define-as --name default --type dir --target /var/lib/libvirt/images
```

```
$ sudo virsh pool-start default
```

```
$ sudo virsh pool-autostart default
```

11. Create a **pull-secret.txt** file:

```
$ vim pull-secret.txt
```

In a web browser, navigate to [Install OpenShift on Bare Metal with installer-provisioned infrastructure](#). Click **Copy pull secret**. Paste the contents into the **pull-secret.txt** file and save the contents in the **kni** user's home directory.

16.3.3. Checking NTP server synchronization

The OpenShift Container Platform installation program installs the **chrony** Network Time Protocol (NTP) service on the cluster nodes. To complete installation, each node must have access to an NTP time server. You can verify NTP server synchronization by using the **chrony** service.

For disconnected clusters, you must configure the NTP servers on the control plane nodes. For more information see the *Additional resources* section.

Prerequisites

- You installed the **chrony** package on the target node.

Procedure

1. Log in to the node by using the **ssh** command.
2. View the NTP servers available to the node by running the following command:

```
$ chronyc sources
```

Example output

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
=====
====
^+ time.cloudflare.com      3 10 377 187 -209us[-209us] +/- 32ms
^+ t1.time.ir2.yahoo.com    2 10 377 185 -4382us[-4382us] +/- 23ms
^+ time.cloudflare.com      3 10 377 198 -996us[-1220us] +/- 33ms
^* brenbox.westnet.ie       1 10 377 193 -9538us[-9761us] +/- 24ms
```

3. Use the **ping** command to ensure that the node can access an NTP server, for example:

```
$ ping time.cloudflare.com
```

Example output

```
PING time.cloudflare.com (162.159.200.123) 56(84) bytes of data.
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=1 ttl=54 time=32.3 ms
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=2 ttl=54 time=30.9 ms
```



```
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=3 ttl=54 time=36.7 ms
```

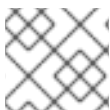
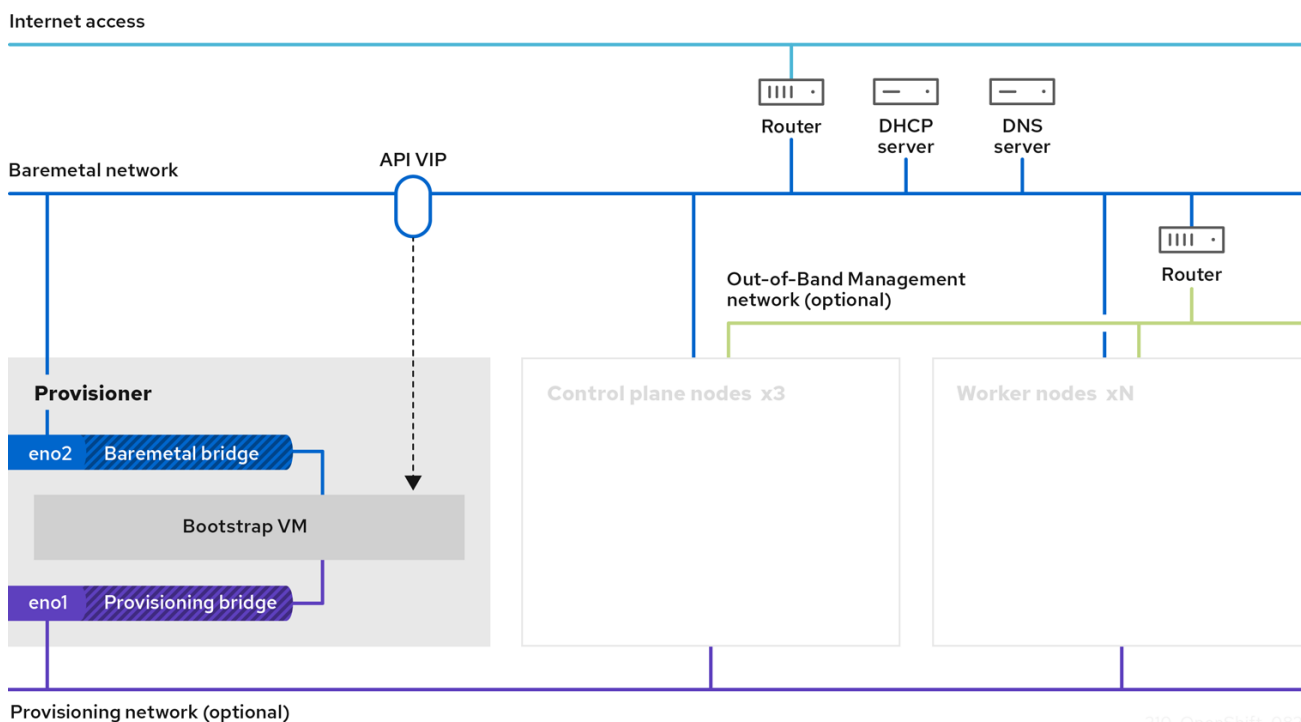
```
...
```

Additional resources

- [Optional: Configuring NTP for disconnected clusters](#)
- [Network Time Protocol \(NTP\)](#)

16.3.4. Configuring networking

Before installation, you must configure the networking on the provisioner node. Installer-provisioned clusters deploy with a bare-metal bridge and network, and an optional provisioning bridge and network.



NOTE

You can also configure networking from the web console.

Procedure

1. Export the bare-metal network NIC name:

```
$ export PUB_CONN=<baremetal_nic_name>
```

2. Configure the bare-metal network:



NOTE

The SSH connection might disconnect after executing these steps.

```
$ sudo nohup bash -c "
  nmcli con down \"\$PUB_CONN\"
  nmcli con delete \"\$PUB_CONN\"
  # RHEL 8.1 appends the word \"System\" in front of the connection, delete in case it exists
  nmcli con down \"System \$PUB_CONN\"
  nmcli con delete \"System \$PUB_CONN\"
  nmcli connection add ifname baremetal type bridge con-name baremetal bridge.stp no
  nmcli con add type bridge-slave ifname \"\$PUB_CONN\" master baremetal
  pkill dhclient;dhclient baremetal
"
```

- Optional: If you are deploying with a provisioning network, export the provisioning network NIC name:

```
$ export PROV_CONN=<prov_nic_name>
```

- Optional: If you are deploying with a provisioning network, configure the provisioning network:

```
$ sudo nohup bash -c "
  nmcli con down \"\$PROV_CONN\"
  nmcli con delete \"\$PROV_CONN\"
  nmcli connection add ifname provisioning type bridge con-name provisioning
  nmcli con add type bridge-slave ifname \"\$PROV_CONN\" master provisioning
  nmcli connection modify provisioning ipv6.addresses fd00:1101::1/64 ipv6.method manual
  nmcli con down provisioning
  nmcli con up provisioning
"
```



NOTE

The ssh connection might disconnect after executing these steps.

The IPv6 address can be any address as long as it is not routable via the bare-metal network.

Ensure that UEFI is enabled and UEFI PXE settings are set to the IPv6 protocol when using IPv6 addressing.

- Optional: If you are deploying with a provisioning network, configure the IPv4 address on the provisioning network connection:

```
$ nmcli connection modify provisioning ipv4.addresses 172.22.0.254/24 ipv4.method manual
```

- ssh** back into the **provisioner** node (if required):

```
# ssh kni@provisioner.<cluster-name>.<domain>
```

- Verify the connection bridges have been properly created:

```
$ sudo nmcli con show
```

NAME	UUID	TYPE	DEVICE
------	------	------	--------

```

baremetal      4d5133a5-8351-4bb9-bfd4-3af264801530 bridge baremetal
provisioning   43942805-017f-4d7d-a2c2-7cb3324482ed bridge provisioning
virbr0        d9bca40f-eee1-410b-8879-a2d4bb0465e7 bridge virbr0
bridge-slave-eno1 76a8ed50-c7e5-4999-b4f6-6d9014dd0812 ethernet eno1
bridge-slave-eno2 f31c3353-54b7-48de-893a-02d2b34c4736 ethernet eno2

```

16.3.5. Establishing communication between subnets

In a typical OpenShift Container Platform cluster setup, all nodes, including the control plane and worker nodes, reside in the same network. However, for edge computing scenarios, it can be beneficial to locate worker nodes closer to the edge. This often involves using different network segments or subnets for the remote worker nodes than the subnet used by the control plane and local worker nodes. Such a setup can reduce latency for the edge and allow for enhanced scalability. However, the network must be configured properly before installing OpenShift Container Platform to ensure that the edge subnets containing the remote worker nodes can reach the subnet containing the control plane nodes and receive traffic from the control plane too.



IMPORTANT

All control plane nodes must run in the same subnet. When using more than one subnet, you can also configure the Ingress VIP to run on the control plane nodes by using a manifest. See "Configuring network components to run on the control plane" for details.

Deploying a cluster with multiple subnets requires using virtual media.

This procedure details the network configuration required to allow the remote worker nodes in the second subnet to communicate effectively with the control plane nodes in the first subnet and to allow the control plane nodes in the first subnet to communicate effectively with the remote worker nodes in the second subnet.

In this procedure, the cluster spans two subnets:

- The first subnet (**10.0.0.0**) contains the control plane and local worker nodes.
- The second subnet (**192.168.0.0**) contains the edge worker nodes.

Procedure

1. Configure the first subnet to communicate with the second subnet:
 - a. Log in as **root** to a control plane node by running the following command:

```
$ sudo su -
```

- b. Get the name of the network interface:

```
# nmcli dev status
```

- c. Add a route to the second subnet (**192.168.0.0**) via the gateway: s+

```
# nmcli connection modify <interface_name> +ipv4.routes "192.168.0.0/24 via <gateway>"
```

+ Replace **<interface_name>** with the interface name. Replace **<gateway>** with the IP address of the actual gateway.

+ .Example

+

```
# nmcli connection modify eth0 +ipv4.routes "192.168.0.0/24 via 192.168.0.1"
```

a. Apply the changes:

```
# nmcli connection up <interface_name>
```

Replace **<interface_name>** with the interface name.

b. Verify the routing table to ensure the route has been added successfully:

```
# ip route
```

c. Repeat the previous steps for each control plane node in the first subnet.



NOTE

Adjust the commands to match your actual interface names and gateway.

1. Configure the second subnet to communicate with the first subnet:

d. Log in as **root** to a remote worker node:

```
$ sudo su -
```

e. Get the name of the network interface:

```
# nmcli dev status
```

f. Add a route to the first subnet (**10.0.0.0**) via the gateway:

```
# nmcli connection modify <interface_name> +ipv4.routes "10.0.0.0/24 via <gateway>"
```

Replace **<interface_name>** with the interface name. Replace **<gateway>** with the IP address of the actual gateway.

Example

```
# nmcli connection modify eth0 +ipv4.routes "10.0.0.0/24 via 10.0.0.1"
```

g. Apply the changes:

```
# nmcli connection up <interface_name>
```

Replace **<interface_name>** with the interface name.

h. Verify the routing table to ensure the route has been added successfully:

```
# ip route
```

- i. Repeat the previous steps for each worker node in the second subnet.



NOTE

Adjust the commands to match your actual interface names and gateway.

1. Once you have configured the networks, test the connectivity to ensure the remote worker nodes can reach the control plane nodes and the control plane nodes can reach the remote worker nodes.
- j. From the control plane nodes in the first subnet, ping a remote worker node in the second subnet:

```
$ ping <remote_worker_node_ip_address>
```

If the ping is successful, it means the control plane nodes in the first subnet can reach the remote worker nodes in the second subnet. If you don't receive a response, review the network configurations and repeat the procedure for the node.

- k. From the remote worker nodes in the second subnet, ping a control plane node in the first subnet:

```
$ ping <control_plane_node_ip_address>
```

If the ping is successful, it means the remote worker nodes in the second subnet can reach the control plane in the first subnet. If you don't receive a response, review the network configurations and repeat the procedure for the node.

16.3.6. Retrieving the OpenShift Container Platform installer

Use the **stable-4.x** version of the installation program and your selected architecture to deploy the generally available stable version of OpenShift Container Platform:

```
$ export VERSION=stable-4.12
```

```
$ export RELEASE_ARCH=<architecture>
```

```
$ export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-
v4/$RELEASE_ARCH/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print
$3}')
```

16.3.7. Extracting the OpenShift Container Platform installer

After retrieving the installer, the next step is to extract it.

Procedure

1. Set the environment variables:

```
$ export cmd=openshift-baremetal-install
```

```
$ export pullsecret_file=~/.pull-secret.txt
```

-

```
$ export extract_dir=$(pwd)
```

2. Get the **oc** binary:

```
$ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

3. Extract the installer:

```
$ sudo cp oc /usr/local/bin
```

```
$ oc adm release extract --registry-config "${pullsecret_file}" --command=$cmd --to "${extract_dir}" ${RELEASE_IMAGE}
```

```
$ sudo cp openshift-baremetal-install /usr/local/bin
```

16.3.8. Optional: Creating an RHCOS images cache

To employ image caching, you must download the Red Hat Enterprise Linux CoreOS (RHCOS) image used by the bootstrap VM to provision the cluster nodes. Image caching is optional, but it is especially useful when running the installation program on a network with limited bandwidth.



NOTE

The installation program no longer needs the **clusterOSImage** RHCOS image because the correct image is in the release payload.

If you are running the installation program on a network with limited bandwidth and the RHCOS images download takes more than 15 to 20 minutes, the installation program will timeout. Caching images on a web server will help in such scenarios.



WARNING

If you enable TLS for the HTTPD server, you must confirm the root certificate is signed by an authority trusted by the client and verify the trusted certificate chain between your OpenShift Container Platform hub and spoke clusters and the HTTPD server. Using a server configured with an untrusted certificate prevents the images from being downloaded to the image creation service. Using untrusted HTTPS servers is not supported.

Install a container that contains the images.

Procedure

1. Install **podman**:

```
$ sudo dnf install -y podman
```

- Open firewall port **8080** to be used for RHCOS image caching:

```
$ sudo firewall-cmd --add-port=8080/tcp --zone=public --permanent
```

```
$ sudo firewall-cmd --reload
```

- Create a directory to store the **bootstraposimage**:

```
$ mkdir /home/kni/rhcos_image_cache
```

- Set the appropriate SELinux context for the newly created directory:

```
$ sudo semanage fcontext -a -t httpd_sys_content_t "/home/kni/rhcos_image_cache(/.*)?"
```

```
$ sudo restorecon -Rv /home/kni/rhcos_image_cache/
```

- Get the URI for the RHCOS image that the installation program will deploy on the bootstrap VM:

```
$ export RHCOS_QEMU_URI=$(/usr/local/bin/openshift-baremetal-install coreos print-stream-json | jq -r --arg ARCH "${arch}" '.architectures[$ARCH].artifacts.qemu.formats["qcow2.gz"].disk.location')
```

- Get the name of the image that the installation program will deploy on the bootstrap VM:

```
$ export RHCOS_QEMU_NAME=${RHCOS_QEMU_URI##*/}
```

- Get the SHA hash for the RHCOS image that will be deployed on the bootstrap VM:

```
$ export RHCOS_QEMU_UNCOMPRESSED_SHA256=$(/usr/local/bin/openshift-baremetal-install coreos print-stream-json | jq -r --arg ARCH "${arch}" '.architectures[$ARCH].artifacts.qemu.formats["qcow2.gz"].disk["uncompressed-sha256"]')
```

- Download the image and place it in the **/home/kni/rhcos_image_cache** directory:

```
$ curl -L ${RHCOS_QEMU_URI} -o /home/kni/rhcos_image_cache/${RHCOS_QEMU_NAME}
```

- Confirm SELinux type is of **httpd_sys_content_t** for the new file:

```
$ ls -Z /home/kni/rhcos_image_cache
```

- Create the pod:

```
$ podman run -d --name rhcos_image_cache \
-v /home/kni/rhcos_image_cache:/var/www/html \
-p 8080:8080/tcp \
registry.access.redhat.com/ubi9/httpd-24
```

- 1 Creates a caching webserver with the name **rhcos_image_cache**. This pod serves the **bootstrapOSImage** image in the **install-config.yaml** file for deployment.

11. Generate the **bootstrapOSImage** configuration:

```
$ export BAREMETAL_IP=$(ip addr show dev baremetal | awk '/inet /{print $2}' | cut -d"/" -f1)
```

```
$ export
BOOTSTRAP_OS_IMAGE="http://${BAREMETAL_IP}:8080/${RHCOS_QEMU_NAME}?
sha256=${RHCOS_QEMU_UNCOMPRESSED_SHA256}"
```

```
$ echo " bootstrapOSImage=${BOOTSTRAP_OS_IMAGE}"
```

12. Add the required configuration to the **install-config.yaml** file under **platform.baremetal**:

```
platform:
  baremetal:
    bootstrapOSImage: <bootstrap_os_image> 1
```

- 1 Replace **<bootstrap_os_image>** with the value of **\$BOOTSTRAP_OS_IMAGE**.

See the "Configuring the install-config.yaml file" section for additional details.

16.3.9. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, hostnames are set through **NetworkManager**. By default, the machines obtain their hostnames through DHCP. If hostnames are not provided by DHCP, set statically through kernel arguments, or another method, they are obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect hostnames as **localhost** or similar. You can avoid this delay in assigning hostnames by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

16.3.10. Configuring the install-config.yaml file

16.3.10.1. Configuring the install-config.yaml file

The **install-config.yaml** file requires some additional details. Most of the information teaches the installation program and the resulting cluster enough about the available hardware that it is able to fully manage it.



NOTE

The installation program no longer needs the **clusterOSImage** RHCOS image because the correct image is in the release payload.

1. Configure **install-config.yaml**. Change the appropriate variables to match the environment, including **pullSecret** and **sshKey**:


```

apiVersion: v1
baseDomain: <domain>
metadata:
  name: <cluster_name>
networking:
  machineNetwork:
    - cidr: <public_cidr>
  networkType: OVNKubernetes
compute:
- name: worker
  replicas: 2 1
controlPlane:
  name: master
  replicas: 3
platform:
  baremetal: {}
platform:
  baremetal:
    apiVIPs:
      - <api_ip>
    ingressVIPs:
      - <wildcard_ip>
    provisioningNetworkCIDR: <CIDR>
    bootstrapExternalStaticIP: <bootstrap_static_ip_address> 2
    bootstrapExternalStaticGateway: <bootstrap_static_gateway> 3
  hosts:
    - name: openshift-master-0
      role: master
      bmc:
        address: ipmi://<out_of_band_ip> 4
        username: <user>
        password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
          deviceName: "<installation_disk_drive_path>" 5
    - name: <openshift_master_1>
      role: master
      bmc:
        address: ipmi://<out_of_band_ip>
        username: <user>
        password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
          deviceName: "<installation_disk_drive_path>"
    - name: <openshift_master_2>
      role: master
      bmc:
        address: ipmi://<out_of_band_ip>
        username: <user>
        password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
          deviceName: "<installation_disk_drive_path>"
    - name: <openshift_worker_0>
      role: worker

```

```

bmc:
  address: ipmi://<out_of_band_ip>
  username: <user>
  password: <password>
  bootMACAddress: <NIC1_mac_address>
- name: <openshift_worker_1>
  role: worker
  bmc:
    address: ipmi://<out_of_band_ip>
    username: <user>
    password: <password>
    bootMACAddress: <NIC1_mac_address>
  rootDeviceHints:
    deviceName: "<installation_disk_drive_path>"
pullSecret: '<pull_secret>'
sshKey: '<ssh_pub_key>'

```

- 1 Scale the worker machines based on the number of worker nodes that are part of the OpenShift Container Platform cluster. Valid options for the **replicas** value are **0** and integers greater than or equal to **2**. Set the number of replicas to **0** to deploy a three-node cluster, which contains only three control plane machines. A three-node cluster is a smaller, more resource-efficient cluster that can be used for testing, development, and production. You cannot install the cluster with only one worker.
- 2 When deploying a cluster with static IP addresses, you must set the **bootstrapExternalStaticIP** configuration setting to specify the static IP address of the bootstrap VM when there is no DHCP server on the bare-metal network.
- 3 When deploying a cluster with static IP addresses, you must set the **bootstrapExternalStaticGateway** configuration setting to specify the gateway IP address for the bootstrap VM when there is no DHCP server on the bare-metal network.
- 4 See the BMC addressing sections for more options.
- 5 To set the path to the installation disk drive, enter the kernel name of the disk. For example, **/dev/sda**.

IMPORTANT

Because the disk discovery order is not guaranteed, the kernel name of the disk can change across booting options for machines with multiple disks. For instance, **/dev/sda** becomes **/dev/sdb** and vice versa. To avoid this issue, you must use persistent disk attributes, such as the disk World Wide Name (WWN). To use the disk WWN, replace the **deviceName** parameter with the **wwnWithExtension** parameter. Depending on the parameter that you use, enter the disk name, for example, **/dev/sda** or the disk WWN, for example, **"0x64cd98f04fde100024684cf3034da5c2"**. Ensure that you enter the disk WWN value within quotes so that it is used as a string value and not a hexadecimal value.

Failure to meet these requirements for the **rootDeviceHints** parameter might result in the following error:

```
ironic-inspector inspection failed: No disks satisfied root device hints
```

**NOTE**

Before OpenShift Container Platform 4.12, the cluster installation program only accepted an IPv4 address or an IPv6 address for the **apiVIP** and **ingressVIP** configuration settings. In OpenShift Container Platform 4.12 and later, these configuration settings are deprecated. Instead, use a list format in the **apiVIPs** and **ingressVIPs** configuration settings to specify IPv4 addresses, IPv6 addresses, or both IP address formats.

2. Create a directory to store the cluster configuration:

```
$ mkdir ~/clusterconfigs
```

3. Copy the **install-config.yaml** file to the new directory:

```
$ cp install-config.yaml ~/clusterconfigs
```

4. Ensure all bare metal nodes are powered off prior to installing the OpenShift Container Platform cluster:

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

5. Remove old bootstrap resources if any are left over from a previous deployment attempt:

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```


16.3.10.2. Additional install-config parameters

See the following tables for the required parameters, the **hosts** parameter, and the **bmc** parameter for the **install-config.yaml** file.

Table 16.4. Required parameters

Parameters	Default	Description
baseDomain		The domain name for the cluster. For example, example.com .
bootMode	UEFI	The boot mode for a node. Options are legacy , UEFI , and UEFISecureBoot . If bootMode is not set, Ironic sets it while inspecting the node.
bootstrapExternalStaticIP		The static IP address for the bootstrap VM. You must set this value when deploying a cluster with static IP addresses when there is no DHCP server on the bare-metal network.

Parameters	Default	Description
bootstrapExternalStaticGateway		The static IP address of the gateway for the bootstrap VM. You must set this value when deploying a cluster with static IP addresses when there is no DHCP server on the bare-metal network.
sshKey		The sshKey configuration setting contains the key in the <code>~/.ssh/id_rsa.pub</code> file required to access the control plane nodes and worker nodes. Typically, this key is from the provisioner node.
pullSecret		The pullSecret configuration setting contains a copy of the pull secret downloaded from the Install OpenShift on Bare Metal page when preparing the provisioner node.
metadata: name:		The name to be given to the OpenShift Container Platform cluster. For example, openshift .
networking: machineNetwork: - cidr:		The public CIDR (Classless Inter-Domain Routing) of the external network. For example, 10.0.0.0/24 .
compute: - name: worker		The OpenShift Container Platform cluster requires a name be provided for worker (or compute) nodes even if there are zero nodes.
compute: replicas: 2		Replicas sets the number of worker (or compute) nodes in the OpenShift Container Platform cluster.
controlPlane: name: master		The OpenShift Container Platform cluster requires a name for control plane (master) nodes.
controlPlane: replicas: 3		Replicas sets the number of control plane (master) nodes included as part of the OpenShift Container Platform cluster.
provisioningNetworkInterface		The name of the network interface on nodes connected to the provisioning network. For OpenShift Container Platform 4.9 and later releases, use the bootMACAddress configuration setting to enable Ironic to identify the IP address of the NIC instead of using the provisioningNetworkInterface configuration setting to identify the name of the NIC.

Parameters	Default	Description
defaultMachinePlatform		The default configuration used for machine pools without a platform configuration.
apiVIPs		<p>(Optional) The virtual IP address for Kubernetes API communication.</p> <p>This setting must either be provided in the install-config.yaml file as a reserved IP from the MachineNetwork or preconfigured in the DNS so that the default name resolves correctly. Use the virtual IP address and not the FQDN when adding a value to the apiVIPs configuration setting in the install-config.yaml file. The primary IP address must be from the IPv4 network when using dual stack networking. If not set, the installation program uses api.<cluster_name>.<base_domain> to derive the IP address from the DNS.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Before OpenShift Container Platform 4.12, the cluster installation program only accepted an IPv4 address or an IPv6 address for the apiVIP configuration setting. From OpenShift Container Platform 4.12 or later, the apiVIP configuration setting is deprecated. Instead, use a list format for the apiVIPs configuration setting to specify an IPv4 address, an IPv6 address or both IP address formats.</p> </div> </div>
disableCertificateVerification	False	redfish and redfish-virtualmedia need this parameter to manage BMC addresses. The value should be True when using a self-signed certificate for BMC addresses.

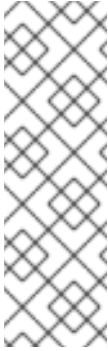
Parameters	Default	Description
ingressVIPs		<p>(Optional) The virtual IP address for ingress traffic.</p> <p>This setting must either be provided in the install-config.yaml file as a reserved IP from the MachineNetwork or preconfigured in the DNS so that the default name resolves correctly. Use the virtual IP address and not the FQDN when adding a value to the ingressVIPs configuration setting in the install-config.yaml file. The primary IP address must be from the IPv4 network when using dual stack networking. If not set, the installation program uses test.apps.<cluster_name>.<base_domain> to derive the IP address from the DNS.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Before OpenShift Container Platform 4.12, the cluster installation program only accepted an IPv4 address or an IPv6 address for the ingressVIP configuration setting. In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a list format for the ingressVIPs configuration setting to specify an IPv4 addresses, an IPv6 addresses or both IP address formats.</p> </div> </div>

Table 16.5. Optional Parameters


Parameters	Default	Description
provisioningDHCPRange	172.22.0.10,172.22.0.100	Defines the IP range for nodes on the provisioning network.
provisioningNetworkCIDR	172.22.0.0/24	The CIDR for the network to use for provisioning. This option is required when not using the default address range on the provisioning network.
clusterProvisioningIP	The third IP address of the provisioningNetworkCIDR .	The IP address within the cluster where the provisioning services run. Defaults to the third IP address of the provisioning subnet. For example, 172.22.0.3 .
bootstrapProvisioningIP	The second IP address of the provisioningNetworkCIDR .	The IP address on the bootstrap VM where the provisioning services run while the installer is deploying the control plane (master) nodes. Defaults to the second IP address of the provisioning subnet. For example, 172.22.0.2 or 2620:52:0:1307::2 .
externalBridge	baremetal	The name of the bare-metal bridge of the hypervisor attached to the bare-metal network.

Parameters	Default	Description
provisioningBridge	provisioning	The name of the provisioning bridge on the provisioner host attached to the provisioning network.
architecture		Defines the host architecture for your cluster. Valid values are amd64 or arm64 .
defaultMachinePlatform		The default configuration used for machine pools without a platform configuration.
bootstrapOSImage		A URL to override the default operating system image for the bootstrap node. The URL must contain a SHA-256 hash of the image. For example: <a href="https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256>">https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256> .
provisioningNetwork		<p>The provisioningNetwork configuration setting determines whether the cluster uses the provisioning network. If it does, the configuration setting also determines if the cluster manages the network.</p> <p>Disabled: Set this parameter to Disabled to disable the requirement for a provisioning network. When set to Disabled, you must only use virtual media based provisioning, or bring up the cluster using the assisted installer. If Disabled and using power management, BMCs must be accessible from the bare-metal network. If Disabled, you must provide two IP addresses on the bare-metal network that are used for the provisioning services.</p> <p>Managed: Set this parameter to Managed, which is the default, to fully manage the provisioning network, including DHCP, TFTP, and so on.</p> <p>Unmanaged: Set this parameter to Unmanaged to enable the provisioning network but take care of manual configuration of DHCP. Virtual media provisioning is recommended but PXE is still available if required.</p>
httpProxy		Set this parameter to the appropriate HTTP proxy used within your environment.
httpsProxy		Set this parameter to the appropriate HTTPS proxy used within your environment.
noProxy		Set this parameter to the appropriate list of exclusions for proxy usage within your environment.

Hosts

The **hosts** parameter is a list of separate bare metal assets used to build the cluster.

Table 16.6. Hosts

Name	Default	Description
name		The name of the BareMetalHost resource to associate with the details. For example, openshift-master-0 .
role		The role of the bare metal node. Either master or worker .
bmc		Connection details for the baseboard management controller. See the BMC addressing section for additional details.
bootMACAddress		<p>The MAC address of the NIC that the host uses for the provisioning network. Ironic retrieves the IP address using the bootMACAddress configuration setting. Then, it binds to the host.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>You must provide a valid MAC address from the host if you disabled the provisioning network.</p> </div> </div>
networkConfig		Set this optional parameter to configure the network interface of a host. See "(Optional) Configuring host network interfaces" for additional details.

16.3.10.3. BMC addressing

Most vendors support Baseboard Management Controller (BMC) addressing with the Intelligent Platform Management Interface (IPMI). IPMI does not encrypt communications. It is suitable for use within a data center over a secured or dedicated management network. Check with your vendor to see if they support Redfish network boot. Redfish delivers simple and secure management for converged, hybrid IT and the Software Defined Data Center (SDDC). Redfish is human readable and machine capable, and leverages common internet and web services standards to expose information directly to the modern tool chain. If your hardware does not support Redfish network boot, use IPMI.

IPMI

Hosts using IPMI use the **ipmi://<out-of-band-ip>:<port>** address format, which defaults to port **623** if not specified. The following example demonstrates an IPMI configuration within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
```



```
address: ipmi://<out-of-band-ip>
username: <user>
password: <password>
```



IMPORTANT

The **provisioning** network is required when PXE booting using IPMI for BMC addressing. It is not possible to PXE boot hosts without a **provisioning** network. If you deploy without a **provisioning** network, you must use a virtual media BMC addressing option such as **redfish-virtualmedia** or **idrac-virtualmedia**. See "Redfish virtual media for HPE iLO" in the "BMC addressing for HPE iLO" section or "Redfish virtual media for Dell iDRAC" in the "BMC addressing for Dell iDRAC" section for additional details.

Redfish network boot

To enable Redfish, use **redfish://** or **redfish+http://** to disable TLS. The installer requires both the hostname or the IP address and the path to the system ID. The following example demonstrates a Redfish configuration within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
      username: <user>
      password: <password>
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include **disableCertificateVerification: True** in the **bmc** configuration if using self-signed certificates. The following example demonstrates a Redfish configuration using the **disableCertificateVerification: True** configuration parameter within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
      username: <user>
      password: <password>
      disableCertificateVerification: True
```

Redfish APIs

Several redfish API endpoints are called onto your BCM when using the bare-metal installer-provisioned infrastructure.



IMPORTANT

You need to ensure that your BMC supports all of the redfish APIs before installation.

List of redfish APIs

- Power on

```
curl -u $USER:$PASS -X POST -H'Content-Type: application/json' -H'Accept:
application/json' -d '{"ResetType": "On"}'
https://$SERVER/redfish/v1/Systems/$SystemID/Actions/ComputerSystem.Reset
```

- Power off

```
curl -u $USER:$PASS -X POST -H'Content-Type: application/json' -H'Accept:
application/json' -d '{"ResetType": "ForceOff"}'
https://$SERVER/redfish/v1/Systems/$SystemID/Actions/ComputerSystem.Reset
```

- Temporary boot using **pxe**

```
curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot": {"BootSourceOverrideTarget":
"pxe", "BootSourceOverrideEnabled": "Once"}}'
```

- Set BIOS boot mode using **Legacy** or **UEFI**

```
curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot":
{"BootSourceOverrideMode": "UEFI"}}'
```

List of redfish-virtualmedia APIs

- Set temporary boot device using **cd** or **dvd**

```
curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot": {"BootSourceOverrideTarget":
"cd", "BootSourceOverrideEnabled": "Once"}}'
```

- Mount virtual media

```
curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json" -H "If-Match: *"
https://$Server/redfish/v1/Managers/$ManagerID/VirtualMedia/$VmediaId -d '{"Image":
"https://example.com/test.iso", "TransferProtocolType": "HTTPS", "UserName": "",
"Password": ""}'
```



NOTE

The **PowerOn** and **PowerOff** commands for redfish APIs are the same for the redfish-virtualmedia APIs.



IMPORTANT

HTTPS and **HTTP** are the only supported parameter types for **TransferProtocolTypes**.

16.3.10.4. BMC addressing for Dell iDRAC

The **address** field for each **bmc** entry is a URL for connecting to the OpenShift Container Platform cluster nodes, including the type of controller in the URL scheme and its location on the network.

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
    bmc:
      address: <address> 1
      username: <user>
      password: <password>
```

1 The **address** configuration setting specifies the protocol.

For Dell hardware, Red Hat supports integrated Dell Remote Access Controller (iDRAC) virtual media, Redfish network boot, and IPMI.

BMC address formats for Dell iDRAC

Protocol	Address Format
iDRAC virtual media	idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
Redfish network boot	redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
IPMI	ipmi://<out-of-band-ip>



IMPORTANT

Use **idrac-virtualmedia** as the protocol for Redfish virtual media. **redfish-virtualmedia** will not work on Dell hardware. Dell's **idrac-virtualmedia** uses the Redfish standard with Dell's OEM extensions.

See the following sections for additional details.

Redfish virtual media for Dell iDRAC

For Redfish virtual media on Dell servers, use **idrac-virtualmedia://** in the **address** setting. Using **redfish-virtualmedia://** will not work.



NOTE

Use **idrac-virtualmedia://** as the protocol for Redfish virtual media. Using **redfish-virtualmedia://** will not work on Dell hardware, because the **idrac-virtualmedia://** protocol corresponds to the **idrac** hardware type and the Redfish protocol in Ironic. Dell's **idrac-virtualmedia://** protocol uses the Redfish standard with Dell's OEM extensions. Ironic also supports the **idrac** type with the WSMAN protocol. Therefore, you must specify **idrac-virtualmedia://** to avoid unexpected behavior when electing to use Redfish with virtual media on Dell hardware.

The following example demonstrates using iDRAC virtual media within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
      username: <user>
      password: <password>
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include **disableCertificateVerification: True** in the **bmc** configuration if using self-signed certificates.



NOTE

Ensure the OpenShift Container Platform cluster nodes have **AutoAttach** enabled through the iDRAC console. The menu path is: **Configuration** → **Virtual Media** → **Attach Mode** → **AutoAttach**.

The following example demonstrates a Redfish configuration using the **disableCertificateVerification: True** configuration parameter within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
      username: <user>
      password: <password>
      disableCertificateVerification: True
```

Redfish network boot for iDRAC

To enable Redfish, use **redfish://** or **redfish+http://** to disable transport layer security (TLS). The installer requires both the hostname or the IP address and the path to the system ID. The following example demonstrates a Redfish configuration within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
      username: <user>
      password: <password>
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include **disableCertificateVerification: True** in the **bmc** configuration if using self-signed certificates. The following example demonstrates a Redfish configuration using the **disableCertificateVerification: True** configuration parameter within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
      username: <user>
      password: <password>
      disableCertificateVerification: True
```



NOTE

There is a known issue on Dell iDRAC 9 with firmware version **04.40.00.00** and all releases up to including the **5.xx** series for installer-provisioned installations on bare metal deployments. The virtual console plugin defaults to eHTML5, an enhanced version of HTML5, which causes problems with the **InsertVirtualMedia** workflow. Set the plugin to use HTML5 to avoid this issue. The menu path is **Configuration** → **Virtual console** → **Plug-in Type** → **HTML5**.

Ensure the OpenShift Container Platform cluster nodes have **AutoAttach** enabled through the iDRAC console. The menu path is: **Configuration** → **Virtual Media** → **Attach Mode** → **AutoAttach**.

16.3.10.5. BMC addressing for HPE iLO

The **address** field for each **bmc** entry is a URL for connecting to the OpenShift Container Platform cluster nodes, including the type of controller in the URL scheme and its location on the network.

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
    bmc:
      address: <address> 1
      username: <user>
      password: <password>
```

1 The **address** configuration setting specifies the protocol.

For HPE integrated Lights Out (iLO), Red Hat supports Redfish virtual media, Redfish network boot, and IPMI.

Table 16.7. BMC address formats for HPE iLO

Protocol	Address Format
Redfish virtual media	redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
Redfish network boot	redfish://<out-of-band-ip>/redfish/v1/Systems/1

Protocol	Address Format
IPMI	ipmi://<out-of-band-ip>

See the following sections for additional details.

Redfish virtual media for HPE iLO

To enable Redfish virtual media for HPE servers, use **redfish-virtualmedia://** in the **address** setting. The following example demonstrates using Redfish virtual media within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
      username: <user>
      password: <password>
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include **disableCertificateVerification: True** in the **bmc** configuration if using self-signed certificates. The following example demonstrates a Redfish configuration using the **disableCertificateVerification: True** configuration parameter within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
      username: <user>
      password: <password>
      disableCertificateVerification: True
```



NOTE

Redfish virtual media is not supported on 9th generation systems running iLO4, because Ironic does not support iLO4 with virtual media.

Redfish network boot for HPE iLO

To enable Redfish, use **redfish://** or **redfish+http://** to disable TLS. The installer requires both the hostname or the IP address and the path to the system ID. The following example demonstrates a Redfish configuration within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
```

```
bmc:
  address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
  username: <user>
  password: <password>
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include **disableCertificateVerification: True** in the **bmc** configuration if using self-signed certificates. The following example demonstrates a Redfish configuration using the **disableCertificateVerification: True** configuration parameter within the **install-config.yaml** file.

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

16.3.10.6. BMC addressing for Fujitsu iRMC

The **address** field for each **bmc** entry is a URL for connecting to the OpenShift Container Platform cluster nodes, including the type of controller in the URL scheme and its location on the network.

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
        bmc:
          address: <address> ①
          username: <user>
          password: <password>
```

① The **address** configuration setting specifies the protocol.

For Fujitsu hardware, Red Hat supports integrated Remote Management Controller (iRMC) and IPMI.

Table 16.8. BMC address formats for Fujitsu iRMC

Protocol	Address Format
iRMC	irmc://<out-of-band-ip>
IPMI	ipmi://<out-of-band-ip>

iRMC

Fujitsu nodes can use **irmc://<out-of-band-ip>** and defaults to port **443**. The following example demonstrates an iRMC configuration within the **install-config.yaml** file.

```

platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
    bmc:
      address: irmc://<out-of-band-ip>
      username: <user>
      password: <password>

```

**NOTE**

Currently Fujitsu supports iRMC S5 firmware version 3.05P and above for installer-provisioned installation on bare metal.

16.3.10.7. Root device hints

The **rootDeviceHints** parameter enables the installer to provision the Red Hat Enterprise Linux CoreOS (RHCOS) image to a particular device. The installer examines the devices in the order it discovers them, and compares the discovered values with the hint values. The installer uses the first discovered device that matches the hint value. The configuration can combine multiple hints, but a device must match all hints for the installer to select it.

Table 16.9. Subfields

Subfield	Description
deviceName	A string containing a Linux device name like /dev/vda . The hint must match the actual value exactly.
hctl	A string containing a SCSI bus address like 0:0:0:0 . The hint must match the actual value exactly.
model	A string containing a vendor-specific device identifier. The hint can be a substring of the actual value.
vendor	A string containing the name of the vendor or manufacturer of the device. The hint can be a substring of the actual value.
serialNumber	A string containing the device serial number. The hint must match the actual value exactly.
minSizeGigabytes	An integer representing the minimum size of the device in gigabytes.
wwn	A string containing the unique storage identifier. The hint must match the actual value exactly.

Subfield	Description
wwnWithExtension	A string containing the unique storage identifier with the vendor extension appended. The hint must match the actual value exactly.
wwnVendorExtension	A string containing the unique vendor storage identifier. The hint must match the actual value exactly.
rotational	A boolean indicating whether the device should be a rotating disk (true) or not (false).

Example usage

```
- name: master-0
  role: master
  bmc:
    address: ipmi://10.10.0.3:6203
    username: admin
    password: redhat
  bootMACAddress: de:ad:be:ef:00:40
  rootDeviceHints:
    deviceName: "/dev/sda"
```

16.3.10.8. Optional: Setting proxy settings

To deploy an OpenShift Container Platform cluster using a proxy, make the following changes to the **install-config.yaml** file.

```
apiVersion: v1
baseDomain: <domain>
proxy:
  httpProxy: http://USERNAME:PASSWORD@proxy.example.com:PORT
  httpsProxy: https://USERNAME:PASSWORD@proxy.example.com:PORT
  noProxy: <WILDCARD_OF_DOMAIN>,<PROVISIONING_NETWORK/CIDR>,
  <BMC_ADDRESS_RANGE/CIDR>
```

The following is an example of **noProxy** with values.

```
noProxy: .example.com,172.22.0.0/24,10.10.0.0/24
```

With a proxy enabled, set the appropriate values of the proxy in the corresponding key/value pair.

Key considerations:

- If the proxy does not have an HTTPS proxy, change the value of **httpsProxy** from **https://** to **http://**.
- If using a provisioning network, include it in the **noProxy** setting, otherwise the installer will fail.

- Set all of the proxy settings as environment variables within the provisioner node. For example, **HTTP_PROXY**, **HTTPS_PROXY**, and **NO_PROXY**.

**NOTE**

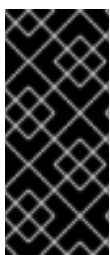
When provisioning with IPv6, you cannot define a CIDR address block in the **noProxy** settings. You must define each address separately.

16.3.10.9. Optional: Deploying with no provisioning network

To deploy an OpenShift Container Platform cluster without a **provisioning** network, make the following changes to the **install-config.yaml** file.

```
platform:
  baremetal:
    apiVIPs:
      - <api_VIP>
    ingressVIPs:
      - <ingress_VIP>
    provisioningNetwork: "Disabled" 1
```

- 1** Add the **provisioningNetwork** configuration setting, if needed, and set it to **Disabled**.

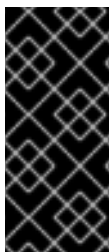
**IMPORTANT**

The **provisioning** network is required for PXE booting. If you deploy without a **provisioning** network, you must use a virtual media BMC addressing option such as **redfish-virtualmedia** or **idrac-virtualmedia**. See "Redfish virtual media for HPE iLO" in the "BMC addressing for HPE iLO" section or "Redfish virtual media for Dell iDRAC" in the "BMC addressing for Dell iDRAC" section for additional details.

16.3.10.10. Optional: Deploying with dual-stack networking

For dual-stack networking in OpenShift Container Platform clusters, you can configure IPv4 and IPv6 address endpoints for cluster nodes. To configure IPv4 and IPv6 address endpoints for cluster nodes, edit the **machineNetwork**, **clusterNetwork**, and **serviceNetwork** configuration settings in the **install-config.yaml** file.

Each setting must have two CIDR entries each. Ensure the first CIDR entry is the IPv4 setting and the second CIDR entry is the IPv6 setting.

**IMPORTANT**

The API VIP IP address and the Ingress VIP address must be of the primary IP address family when using dual-stack networking. Currently, Red Hat does not support dual-stack VIPs or dual-stack networking with IPv6 as the primary IP address family. However, Red Hat does support dual-stack networking with IPv4 as the primary IP address family. Therefore, the IPv4 entries must go before the IPv6 entries.

```
machineNetwork:
  - cidr: {{ extcidrnet }}
  - cidr: {{ extcidrnet6 }}
```

```

clusterNetwork:
- cidr: 10.128.0.0/14
  hostPrefix: 23
- cidr: fd02::/48
  hostPrefix: 64
serviceNetwork:
- 172.30.0.0/16
- fd03::/112

```

IMPORTANT

On a bare-metal platform, if you specified an NMState configuration in the **networkConfig** section of your **install-config.yaml** file, add **interfaces.wait-ip: ipv4+ipv6** to the NMState YAML file to resolve an issue that prevents your cluster from deploying on a dual-stack network.

Example NMState YAML configuration file that includes the **wait-ip** parameter

```

networkConfig:
  nmstate:
    interfaces:
      - name: <interface_name>
      # ...
      wait-ip: ipv4+ipv6
      # ...

```

To provide an interface to the cluster for applications that use IPv4 and IPv6 addresses, configure IPv4 and IPv6 virtual IP (VIP) address endpoints for the Ingress VIP and API VIP services. To configure IPv4 and IPv6 address endpoints, edit the **apiVIPs** and **ingressVIPs** configuration settings in the **install-config.yaml** file. The **apiVIPs** and **ingressVIPs** configuration settings use a list format. The order of the list indicates the primary and secondary VIP address for each service.

```

platform:
  baremetal:
    apiVIPs:
      - <api_ipv4>
      - <api_ipv6>
    ingressVIPs:
      - <wildcard_ipv4>
      - <wildcard_ipv6>

```

16.3.10.11. Optional: Configuring host network interfaces

Before installation, you can set the **networkConfig** configuration setting in the **install-config.yaml** file to configure host network interfaces using NMState.

The most common use case for this functionality is to specify a static IP address on the bare-metal network, but you can also configure other networks such as a storage network. This functionality supports other NMState features such as VLAN, VXLAN, bridges, bonds, routes, MTU, and DNS resolver settings.

Prerequisites

- Configure a **PTR** DNS record with a valid hostname for each node with a static IP address.
- Install the NMState CLI (**nmstate**).

Procedure

1. Optional: Consider testing the NMState syntax with **nmstatectl gc** before including it in the **install-config.yaml** file, because the installer will not check the NMState YAML syntax.



NOTE

Errors in the YAML syntax might result in a failure to apply the network configuration. Additionally, maintaining the validated YAML syntax is useful when applying changes using Kubernetes NMState after deployment or when expanding the cluster.

- a. Create an NMState YAML file:

```

interfaces:
- name: <nic1_name> 1
  type: ethernet
  state: up
  ipv4:
    address:
      - ip: <ip_address> 2
        prefix-length: 24
      enabled: true
  dns-resolver:
    config:
      server:
        - <dns_ip_address> 3
  routes:
    config:
      - destination: 0.0.0.0/0
        next-hop-address: <next_hop_ip_address> 4
        next-hop-interface: <next_hop_nic1_name> 5

```

- 1 2 3 4 5 Replace **<nic1_name>**, **<ip_address>**, **<dns_ip_address>**, **<next_hop_ip_address>** and **<next_hop_nic1_name>** with appropriate values.

- b. Test the configuration file by running the following command:

```
$ nmstatectl gc <nmstate_yaml_file>
```

Replace **<nmstate_yaml_file>** with the configuration file name.

2. Use the **networkConfig** configuration setting by adding the NMState configuration to hosts within the **install-config.yaml** file:

```

hosts:
- name: openshift-master-0
  role: master

```

```

bmc:
  address: redfish+http://<out_of_band_ip>/redfish/v1/Systems/
  username: <user>
  password: <password>
  disableCertificateVerification: null
bootMACAddress: <NIC1_mac_address>
bootMode: UEFI
rootDeviceHints:
  deviceName: "/dev/sda"
networkConfig: ❶
  interfaces:
  - name: <nic1_name> ❷
    type: ethernet
    state: up
    ipv4:
      address:
      - ip: <ip_address> ❸
        prefix-length: 24
      enabled: true
  dns-resolver:
    config:
      server:
      - <dns_ip_address> ❹
  routes:
    config:
    - destination: 0.0.0.0/0
      next-hop-address: <next_hop_ip_address> ❺
      next-hop-interface: <next_hop_nic1_name> ❻

```

❶ Add the NMState YAML syntax to configure the host interfaces.

❷ ❸ ❹ ❺ ❻ Replace <nic1_name>, <ip_address>, <dns_ip_address>, <next_hop_ip_address> and <next_hop_nic1_name> with appropriate values.



IMPORTANT

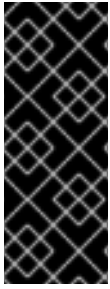
After deploying the cluster, you cannot modify the **networkConfig** configuration setting of **install-config.yaml** file to make changes to the host network interface. Use the Kubernetes NMState Operator to make changes to the host network interface after deployment.

16.3.10.12. Configuring host network interfaces for subnets

For edge computing scenarios, it can be beneficial to locate worker nodes closer to the edge. To locate remote worker nodes in subnets, you might use different network segments or subnets for the remote worker nodes than you used for the control plane subnet and local worker nodes. You can reduce latency for the edge and allow for enhanced scalability by setting up subnets for edge computing scenarios.

If you have established different network segments or subnets for remote worker nodes as described in the section on "Establishing communication between subnets", you must specify the subnets in the **machineNetwork** configuration setting if the workers are using static IP addresses, bonds or other advanced networking. When setting the node IP address in the **networkConfig** parameter for each remote worker node, you must also specify the gateway and the DNS server for the subnet containing

the control plane nodes when using static IP addresses. This ensures the remote worker nodes can reach the subnet containing the control plane nodes and that they can receive network traffic from the control plane.



IMPORTANT

All control plane nodes must run in the same subnet. When using more than one subnet, you can also configure the Ingress VIP to run on the control plane nodes by using a manifest. See "Configuring network components to run on the control plane" for details.

Deploying a cluster with multiple subnets requires using virtual media, such as **redfish-virtualmedia** and **idrac-virtualmedia**.

Procedure

1. Add the subnets to the **machineNetwork** in the **install-config.yaml** file when using static IP addresses:

```
networking:
  machineNetwork:
    - cidr: 10.0.0.0/24
    - cidr: 192.168.0.0/24
  networkType: OVNKubernetes
```

2. Add the gateway and DNS configuration to the **networkConfig** parameter of each edge worker node using NMState syntax when using a static IP address or advanced networking such as bonds:

```
networkConfig:
  interfaces:
    - name: <interface_name> ❶
      type: ethernet
      state: up
      ipv4:
        enabled: true
        dhcp: false
        address:
          - ip: <node_ip> ❷
            prefix-length: 24
        gateway: <gateway_ip> ❸
      dns-resolver:
        config:
          server:
            - <dns_ip> ❹
```

- ❶ Replace **<interface_name>** with the interface name.
- ❷ Replace **<node_ip>** with the IP address of the node.
- ❸ Replace **<gateway_ip>** with the IP address of the gateway.
- ❹ Replace **<dns_ip>** with the IP address of the DNS server.

16.3.10.13. Optional: Configuring address generation modes for SLAAC in dual-stack networks

For dual-stack clusters that use Stateless Address AutoConfiguration (SLAAC), you must specify a global value for the **ipv6.addr-gen-mode** network setting. You can set this value using NMState to configure the ramdisk and the cluster configuration files. If you don't configure a consistent **ipv6.addr-gen-mode** in these locations, IPv6 address mismatches can occur between CSR resources and **BareMetalHost** resources in the cluster.

Prerequisites

- Install the NMState CLI (**nmstate**).

Procedure

1. Optional: Consider testing the NMState YAML syntax with the **nmstatectl gc** command before including it in the **install-config.yaml** file because the installation program will not check the NMState YAML syntax.

- a. Create an NMState YAML file:

```
interfaces:
- name: eth0
  ipv6:
    addr-gen-mode: <address_mode> 1
```

- 1 Replace **<address_mode>** with the type of address generation mode required for IPv6 addresses in the cluster. Valid values are **eui64**, **stable-privacy**, or **random**.

- b. Test the configuration file by running the following command:

```
$ nmstatectl gc <nmstate_yaml_file> 1
```

- 1 Replace **<nmstate_yaml_file>** with the name of the test configuration file.

2. Add the NMState configuration to the **hosts.networkConfig** section within the **install-config.yaml** file:

```
hosts:
- name: openshift-master-0
  role: master
  bmc:
    address: redfish+http://<out_of_band_ip>/redfish/v1/Systems/
    username: <user>
    password: <password>
    disableCertificateVerification: null
  bootMACAddress: <NIC1_mac_address>
  bootMode: UEFI
  rootDeviceHints:
    deviceName: "/dev/sda"
  networkConfig:
    interfaces:
      - name: eth0
```

```

|   ipv6:
|     addr-gen-mode: <address_mode> ❶
|   ...

```

- ❶ Replace **<address_mode>** with the type of address generation mode required for IPv6 addresses in the cluster. Valid values are **eui64**, **stable-privacy**, or **random**.

16.3.10.14. Configuring multiple cluster nodes

You can simultaneously configure OpenShift Container Platform cluster nodes with identical settings. Configuring multiple cluster nodes avoids adding redundant information for each node to the **install-config.yaml** file. This file contains specific parameters to apply an identical configuration to multiple nodes in the cluster.

Compute nodes are configured separately from the controller node. However, configurations for both node types use the highlighted parameters in the **install-config.yaml** file to enable multi-node configuration. Set the **networkConfig** parameters to **BOND**, as shown in the following example:

```

hosts:
- name: otest-master-0
  [...]
  networkConfig: &BOND
  interfaces:
  - name: bond0
    type: bond
    state: up
    ipv4:
      dhcp: true
      enabled: true
    link-aggregation:
      mode: active-backup
      port:
      - enp2s0
      - enp3s0
- name: otest-master-1
  [...]
  networkConfig: *BOND
- name: otest-master-2
  [...]
  networkConfig: *BOND

```



NOTE

Configuration of multiple cluster nodes is only available for initial deployments on installer-provisioned infrastructure.

16.3.10.15. Optional: Configuring managed Secure Boot

You can enable managed Secure Boot when deploying an installer-provisioned cluster using Redfish BMC addressing, such as **redfish**, **redfish-virtualmedia**, or **idrac-virtualmedia**. To enable managed Secure Boot, add the **bootMode** configuration setting to each node:

Example

-


```

hosts:
- name: openshift-master-0
  role: master
  bmc:
    address: redfish://<out_of_band_ip> ❶
    username: <username>
    password: <password>
    bootMACAddress: <NIC1_mac_address>
  rootDeviceHints:
    deviceName: "/dev/sda"
  bootMode: UEFISecureBoot ❷

```

- ❶ Ensure the **bmc.address** setting uses **redfish**, **redfish-virtualmedia**, or **idrac-virtualmedia** as the protocol. See "BMC addressing for HPE iLO" or "BMC addressing for Dell iDRAC" for additional details.
- ❷ The **bootMode** setting is **UEFI** by default. Change it to **UEFISecureBoot** to enable managed Secure Boot.

**NOTE**

See "Configuring nodes" in the "Prerequisites" to ensure the nodes can support managed Secure Boot. If the nodes do not support managed Secure Boot, see "Configuring nodes for Secure Boot manually" in the "Configuring nodes" section. Configuring Secure Boot manually requires Redfish virtual media.

**NOTE**

Red Hat does not support Secure Boot with IPMI, because IPMI does not provide Secure Boot management facilities.

16.3.11. Manifest configuration files

16.3.11.1. Creating the OpenShift Container Platform manifests

1. Create the OpenShift Container Platform manifests.

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
```

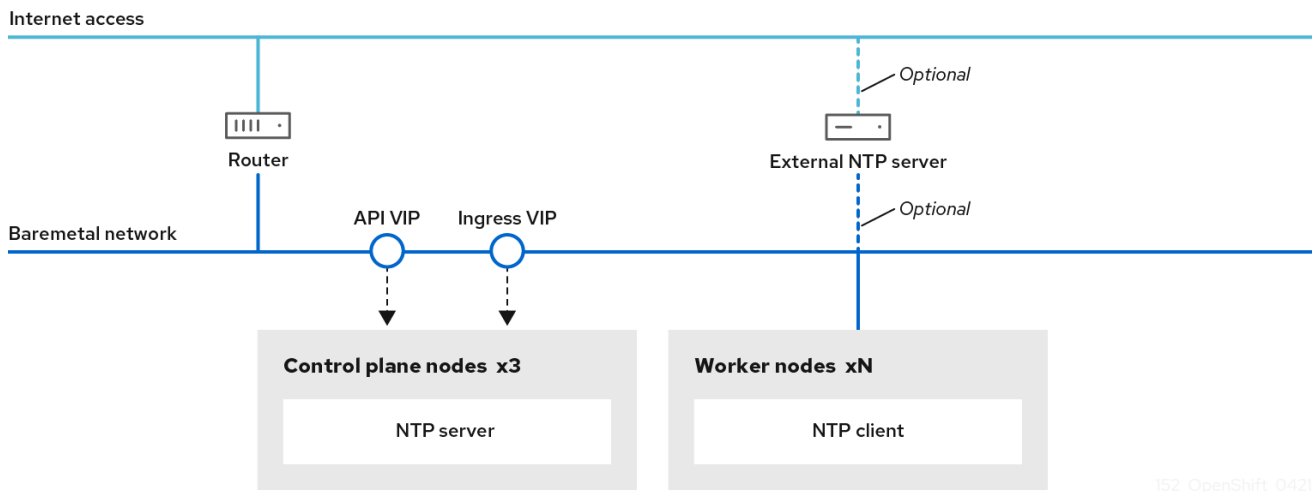
```

INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for Scheduler cluster settings
WARNING Discarding the OpenShift Manifest that was provided in the target directory because its dependencies are dirty and it needs to be regenerated

```

16.3.11.2. Optional: Configuring NTP for disconnected clusters

OpenShift Container Platform installs the **chrony** Network Time Protocol (NTP) service on the cluster nodes.



152_OpenShift_0421

OpenShift Container Platform nodes must agree on a date and time to run properly. When worker nodes retrieve the date and time from the NTP servers on the control plane nodes, it enables the installation and operation of clusters that are not connected to a routable network and thereby do not have access to a higher stratum NTP server.

Procedure

1. Create a Butane config, **99-master-chrony-conf-override.bu**, including the contents of the **chrony.conf** file for the control plane nodes.



NOTE

See "Creating machine configs with Butane" for information about Butane.

Butane config example

```
variant: openshift
version: 4.12.0
metadata:
  name: 99-master-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # Use public servers from the pool.ntp.org project.
          # Please consider joining the pool (https://www.pool.ntp.org/join.html).

          # The Machine Config Operator manages this file
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0
```

```

driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony

# Configure the control plane nodes to serve as local NTP servers
# for all worker nodes, even if they are not in sync with an
# upstream NTP server.

# Allow NTP client access from the local network.
allow all
# Serve time even if not synchronized to a time source.
local stratum 3 orphan

```

- 1 You must replace **<cluster-name>** with the name of the cluster and replace **<domain>** with the fully qualified domain name.
2. Use Butane to generate a **MachineConfig** object file, **99-master-chrony-conf-override.yaml**, containing the configuration to be delivered to the control plane nodes:

```
$ butane 99-master-chrony-conf-override.bu -o 99-master-chrony-conf-override.yaml
```

3. Create a Butane config, **99-worker-chrony-conf-override.bu**, including the contents of the **chrony.conf** file for the worker nodes that references the NTP servers on the control plane nodes.

Butane config example

```

variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # The Machine Config Operator manages this file.
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0

```

```
driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony
```

1 You must replace **<cluster-name>** with the name of the cluster and replace **<domain>** with the fully qualified domain name.

4. Use Butane to generate a **MachineConfig** object file, **99-worker-chrony-conf-override.yaml**, containing the configuration to be delivered to the worker nodes:

```
$ butane 99-worker-chrony-conf-override.bu -o 99-worker-chrony-conf-override.yaml
```

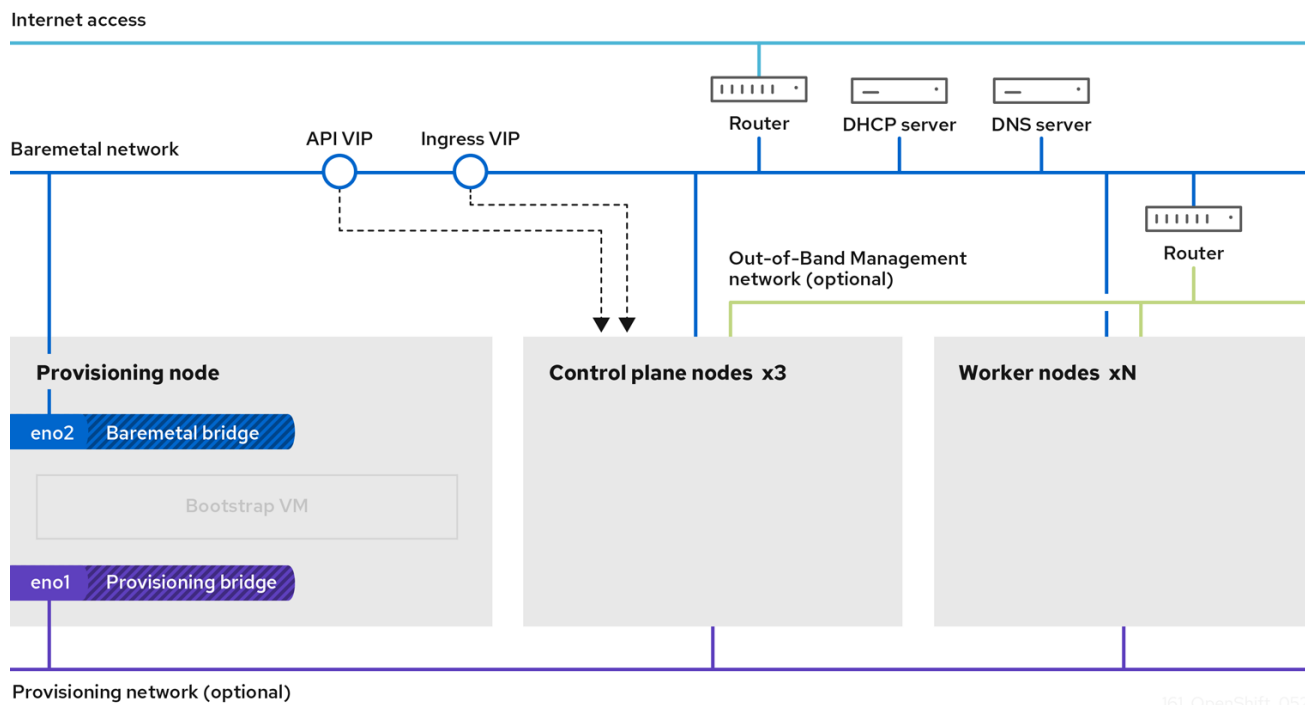
16.3.11.3. Configuring network components to run on the control plane

You can configure networking components to run exclusively on the control plane nodes. By default, OpenShift Container Platform allows any node in the machine config pool to host the **ingressVIP** virtual IP address. However, some environments deploy worker nodes in separate subnets from the control plane nodes, which requires configuring the **ingressVIP** virtual IP address to run on the control plane nodes.



IMPORTANT

When deploying remote workers in separate subnets, you must place the **ingressVIP** virtual IP address exclusively with the control plane nodes.



161_OpenShift_0521

Procedure

1. Change to the directory storing the **install-config.yaml** file:

```
$ cd ~/clusterconfigs
```

2. Switch to the **manifests** subdirectory:

```
$ cd manifests
```

3. Create a file named **cluster-network-avoid-workers-99-config.yaml**:

```
$ touch cluster-network-avoid-workers-99-config.yaml
```

4. Open the **cluster-network-avoid-workers-99-config.yaml** file in an editor and enter a custom resource (CR) that describes the Operator configuration:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: 50-worker-fix-ipi-rwn
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
        - path: /etc/kubernetes/manifests/keepalived.yaml
```

```
mode: 0644
contents:
  source: data,;
```

This manifest places the **ingressVIP** virtual IP address on the control plane nodes. Additionally, this manifest deploys the following processes on the control plane nodes only:

- **openshift-ingress-operator**
- **keepalived**

5. Save the **cluster-network-avoid-workers-99-config.yaml** file.
6. Create a **manifests/cluster-ingress-default-ingresscontroller.yaml** file:

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  nodePlacement:
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/master: ""
```

7. Consider backing up the **manifests** directory. The installer deletes the **manifests/** directory when creating the cluster.
8. Modify the **cluster-scheduler-02-config.yml** manifest to make the control plane nodes schedulable by setting the **mastersSchedulable** field to **true**. Control plane nodes are not schedulable by default. For example:

```
$ sed -i "s;mastersSchedulable: false;mastersSchedulable: true;g"
clusterconfigs/manifests/cluster-scheduler-02-config.yml
```

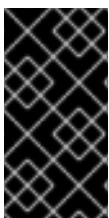


NOTE

If control plane nodes are not schedulable after completing this procedure, deploying the cluster will fail.

16.3.11.4. Optional: Deploying routers on worker nodes

During installation, the installer deploys router pods on worker nodes. By default, the installer installs two router pods. If a deployed cluster requires additional routers to handle external traffic loads destined for services within the OpenShift Container Platform cluster, you can create a **yaml** file to set an appropriate number of router replicas.



IMPORTANT

Deploying a cluster with only one worker node is not supported. While modifying the router replicas will address issues with the **degraded** state when deploying with one worker, the cluster loses high availability for the ingress API, which is not suitable for production environments.

**NOTE**

By default, the installer deploys two routers. If the cluster has no worker nodes, the installer deploys the two routers on the control plane nodes by default.

Procedure

1. Create a **router-replicas.yaml** file:

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  replicas: <num-of-router-pods>
  endpointPublishingStrategy:
    type: HostNetwork
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/worker: ""
```

**NOTE**

Replace **<num-of-router-pods>** with an appropriate value. If working with just one worker node, set **replicas:** to **1**. If working with more than 3 worker nodes, you can increase **replicas:** from the default value **2** as appropriate.

2. Save and copy the **router-replicas.yaml** file to the **clusterconfigs/openshift** directory:

```
$ cp ~/router-replicas.yaml clusterconfigs/openshift/99_router-replicas.yaml
```

16.3.11.5. Optional: Configuring the BIOS

The following procedure configures the BIOS during the installation process.

Procedure

1. Create the manifests.
2. Modify the **BareMetalHost** resource file corresponding to the node:

```
$ vim clusterconfigs/openshift/99_openshift-cluster-api_hosts-*.yaml
```

3. Add the BIOS configuration to the **spec** section of the **BareMetalHost** resource:

```
spec:
  firmware:
    simultaneousMultithreadingEnabled: true
    sriovEnabled: true
    virtualizationEnabled: true
```

**NOTE**

Red Hat supports three BIOS configurations. Only servers with BMC type **irmc** are supported. Other types of servers are currently not supported.

4. Create the cluster.

Additional resources

- [Bare metal configuration](#)

16.3.11.6. Optional: Configuring the RAID

The following procedure configures a redundant array of independent disks (RAID) during the installation process.

**NOTE**

1. OpenShift Container Platform supports hardware RAID for baseboard management controllers (BMCs) using the iRMC protocol only. OpenShift Container Platform 4.12 does not support software RAID.
2. If you want to configure a hardware RAID for the node, verify that the node has a RAID controller.

Procedure

1. Create the manifests.
2. Modify the **BareMetalHost** resource corresponding to the node:

```
$ vim clusterconfigs/openshift/99_openshift-cluster-api_hosts-*.yaml
```

**NOTE**

The following example uses a hardware RAID configuration because OpenShift Container Platform 4.12 does not support software RAID.

- a. If you added a specific RAID configuration to the **spec** section, this causes the node to delete the original RAID configuration in the **preparing** phase and perform a specified configuration on the RAID. For example:

```
spec:
  raid:
    hardwareRAIDVolumes:
      - level: "0" 1
        name: "sda"
        numberOfPhysicalDisks: 1
        rotational: true
        sizeGibibytes: 0
```

- 1** **level** is a required field, and the others are optional fields.

- b. If you added an empty RAID configuration to the **spec** section, the empty configuration causes the node to delete the original RAID configuration during the **preparing** phase, but does not perform a new configuration. For example:

```
spec:
  raid:
    hardwareRAIDVolumes: []
```

- c. If you do not add a **raid** field in the **spec** section, the original RAID configuration is not deleted, and no new configuration will be performed.
3. Create the cluster.

Additional resources

- [Bare metal configuration](#)

16.3.12. Creating a disconnected registry

In some cases, you might want to install an OpenShift Container Platform cluster using a local copy of the installation registry. This could be for enhancing network efficiency because the cluster nodes are on a network that does not have access to the internet.

A local, or mirrored, copy of the registry requires the following:

- A certificate for the registry node. This can be a self-signed certificate.
- A web server that a container on a system will serve.
- An updated pull secret that contains the certificate and local repository information.



NOTE

Creating a disconnected registry on a registry node is optional. If you need to create a disconnected registry on a registry node, you must complete all of the following sub-sections.

Prerequisites

- If you have already prepared a mirror registry for [Mirroring images for a disconnected installation](#), you can skip directly to [Modify the install-config.yaml file to use the disconnected registry](#).

16.3.12.1. Preparing the registry node to host the mirrored registry

The following steps must be completed prior to hosting a mirrored registry on bare metal.

Procedure

1. Open the firewall port on the registry node:

```
$ sudo firewall-cmd --add-port=5000/tcp --zone=libvirt --permanent
```

```
$ sudo firewall-cmd --add-port=5000/tcp --zone=public --permanent
```

-

```
$ sudo firewall-cmd --reload
```

2. Install the required packages for the registry node:

```
$ sudo yum -y install python3 podman httpd httpd-tools jq
```

3. Create the directory structure where the repository information will be held:

```
$ sudo mkdir -p /opt/registry/{auth,certs,data}
```

16.3.12.2. Mirroring the OpenShift Container Platform image repository for a disconnected registry

Complete the following steps to mirror the OpenShift Container Platform image repository for a disconnected registry.

Prerequisites

- Your mirror host has access to the internet.
- You configured a mirror registry to use in your restricted network and can access the certificate and credentials that you configured.
- You downloaded the [pull secret from the Red Hat OpenShift Cluster Manager](#) and modified it to include authentication to your mirror repository.

Procedure

1. Review the [OpenShift Container Platform downloads page](#) to determine the version of OpenShift Container Platform that you want to install and determine the corresponding tag on the [Repository Tags](#) page.
2. Set the required environment variables:

- a. Export the release version:

```
$ OCP_RELEASE=<release_version>
```

For **<release_version>**, specify the tag that corresponds to the version of OpenShift Container Platform to install, such as **4.5.4**.

- b. Export the local registry name and host port:

```
$ LOCAL_REGISTRY='<local_registry_host_name>:<local_registry_host_port>'
```

For **<local_registry_host_name>**, specify the registry domain name for your mirror repository, and for **<local_registry_host_port>**, specify the port that it serves content on.

- c. Export the local repository name:

```
$ LOCAL_REPOSITORY='<local_repository_name>'
```

For **<local_repository_name>**, specify the name of the repository to create in your registry, such as **ocp4/openshift4**.

- d. Export the name of the repository to mirror:

```
$ PRODUCT_REPO='openshift-release-dev'
```

For a production release, you must specify **openshift-release-dev**.

- e. Export the path to your registry pull secret:

```
$ LOCAL_SECRET_JSON='<path_to_pull_secret>'
```

For **<path_to_pull_secret>**, specify the absolute path to and file name of the pull secret for your mirror registry that you created.

- f. Export the release mirror:

```
$ RELEASE_NAME="ocp-release"
```

For a production release, you must specify **ocp-release**.

- g. Export the type of architecture for your server, such as **x86_64**:

```
$ ARCHITECTURE=<server_architecture>
```

- h. Export the path to the directory to host the mirrored images:

```
$ REMOVABLE_MEDIA_PATH=<path> 1
```

- 1** Specify the full path, including the initial forward slash (/) character.

3. Mirror the version images to the mirror registry:

- If your mirror host does not have internet access, take the following actions:
 - i. Connect the removable media to a system that is connected to the internet.
 - ii. Review the images and configuration manifests to mirror:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
  --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
  ${ARCHITECTURE} \
  --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
  --to-release-
  image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
  ${ARCHITECTURE} --dry-run
```

- iii. Record the entire **imageContentSources** section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the **imageContentSources** section to the **install-config.yaml** file during installation.
- iv. Mirror the images to a directory on the removable media:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE}
```

- v. Take the media to the restricted network environment and upload the images to the local container registry.

```
$ oc image mirror -a ${LOCAL_SECRET_JSON} --from-
dir=${REMOVABLE_MEDIA_PATH}/mirror
"file://openshift/release:${OCP_RELEASE}*"
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} 1
```

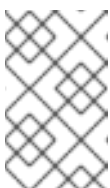
- 1** For **REMOVABLE_MEDIA_PATH**, you must use the same path that you specified when you mirrored the images.

- If the local container registry is connected to the mirror host, take the following actions:
 - i. Directly push the release images to the local registry by using following command:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
--from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
--to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
--to-release-
image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}
```

This command pulls the release information as a digest, and its output includes the **imageContentSources** data that you require when you install your cluster.

- ii. Record the entire **imageContentSources** section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the **imageContentSources** section to the **install-config.yaml** file during installation.



NOTE

The image name gets patched to Quay.io during the mirroring process, and the podman images will show Quay.io in the registry on the bootstrap virtual machine.

4. To create the installation program that is based on the content that you mirrored, extract it and pin it to the release:
 - If your mirror host does not have internet access, run the following command:

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-
baremetal-install "${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}"
```

- If the local container registry is connected to the mirror host, run the following command:

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-
baremetal-install "${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}"
```



IMPORTANT

To ensure that you use the correct images for the version of OpenShift Container Platform that you selected, you must extract the installation program from the mirrored content.

You must perform this step on a machine with an active internet connection.

If you are in a disconnected environment, use the **--image** flag as part of `must-gather` and point to the payload image.

- For clusters using installer-provisioned infrastructure, run the following command:

```
$ openshift-baremetal-install
```

16.3.12.3. Modify the `install-config.yaml` file to use the disconnected registry

On the provisioner node, the `install-config.yaml` file should use the newly created pull-secret from the `pull-secret-update.txt` file. The `install-config.yaml` file must also contain the disconnected registry node's certificate and registry information.

Procedure

- Add the disconnected registry node's certificate to the `install-config.yaml` file:

```
$ echo "additionalTrustBundle: |" >> install-config.yaml
```

The certificate should follow the `"additionalTrustBundle: |"` line and be properly indented, usually by two spaces.

```
$ sed -e 's/^ / /opt/registry/certs/domain.crt >> install-config.yaml
```

- Add the mirror information for the registry to the `install-config.yaml` file:

```
$ echo "imageContentSources:" >> install-config.yaml
```

```
$ echo "- mirrors:" >> install-config.yaml
```

```
$ echo "  - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
```

Replace **registry.example.com** with the registry's fully qualified domain name.

```
$ echo "    source: quay.io/openshift-release-dev/ocp-release" >> install-config.yaml
```

```
$ echo "- mirrors:" >> install-config.yaml
```

```
$ echo " - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
```

Replace **registry.example.com** with the registry's fully qualified domain name.

```
$ echo " source: quay.io/openshift-release-dev/ocp-v4.0-art-dev" >> install-config.yaml
```

16.3.13. Validation checklist for installation

- OpenShift Container Platform installer has been retrieved.
- OpenShift Container Platform installer has been extracted.
- Required parameters for the **install-config.yaml** have been configured.
- The **hosts** parameter for the **install-config.yaml** has been configured.
- The **bmc** parameter for the **install-config.yaml** has been configured.
- Conventions for the values configured in the **bmc address** field have been applied.
- Created the OpenShift Container Platform manifests.
- (Optional) Deployed routers on worker nodes.
- (Optional) Created a disconnected registry.
- (Optional) Validate disconnected registry settings if in use.

16.3.14. Deploying the cluster via the OpenShift Container Platform installer

Run the OpenShift Container Platform installer:

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

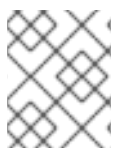
16.3.15. Following the installation

During the deployment process, you can check the installation's overall status by issuing the **tail** command to the **.openshift_install.log** log file in the install directory folder:

```
$ tail -f /path/to/install-dir/.openshift_install.log
```

16.3.16. Verifying static IP address configuration

If the DHCP reservation for a cluster node specifies an infinite lease, after the installer successfully provisions the node, the dispatcher script checks the node's network configuration. If the script determines that the network configuration contains an infinite DHCP lease, it creates a new connection using the IP address of the DHCP lease as a static IP address.



NOTE

The dispatcher script might run on successfully provisioned nodes while the provisioning of other nodes in the cluster is ongoing.

Verify the network configuration is working properly.

Procedure

1. Check the network interface configuration on the node.
2. Turn off the DHCP server and reboot the OpenShift Container Platform node and ensure that the network configuration works properly.

16.3.17. Preparing to reinstall a cluster on bare metal

Before you reinstall a cluster on bare metal, you must perform cleanup operations.

Procedure

1. Remove or reformat the disks for the bootstrap, control plane node, and worker nodes. If you are working in a hypervisor environment, you must add any disks you removed.
2. Delete the artifacts that the previous installation generated:

```
$ cd ; /bin/rm -rf auth/ bootstrap.ign master.ign worker.ign metadata.json \
.openshift_install.log .openshift_install_state.json
```
3. Generate new manifests and Ignition config files. See "Creating the Kubernetes manifest and Ignition config files" for more information.
4. Upload the new bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. This will overwrite the previous Ignition files.

16.3.18. Additional resources

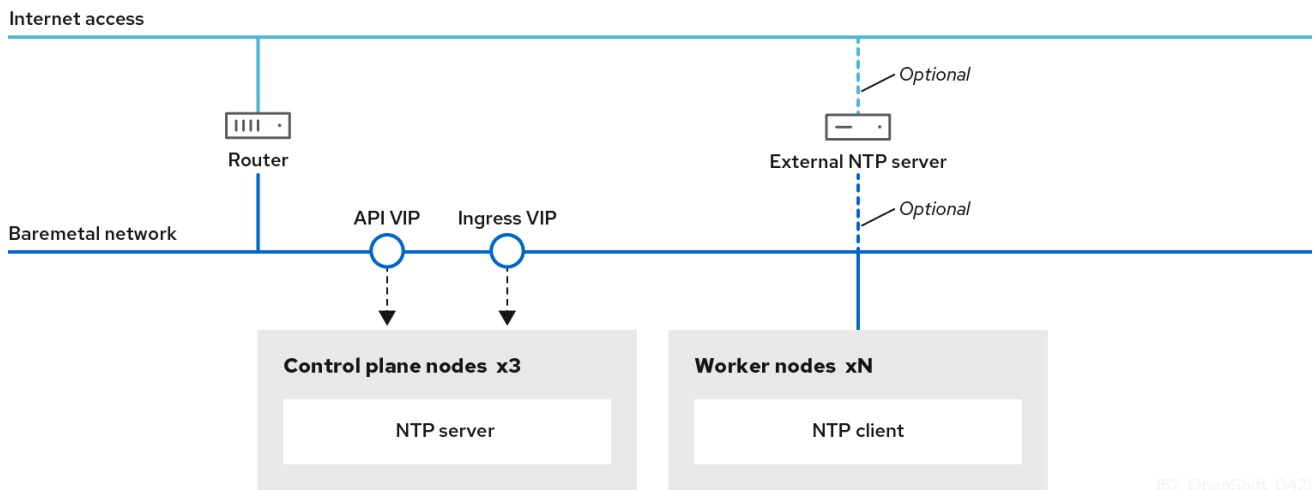
- [OpenShift Container Platform Creating the Kubernetes manifest and Ignition config files](#)
- [Understanding update channels and releases](#)

16.4. INSTALLER-PROVISIONED POSTINSTALLATION CONFIGURATION

After successfully deploying an installer-provisioned cluster, consider the following postinstallation procedures.

16.4.1. Optional: Configuring NTP for disconnected clusters

OpenShift Container Platform installs the **chrony** Network Time Protocol (NTP) service on the cluster nodes. Use the following procedure to configure NTP servers on the control plane nodes and configure worker nodes as NTP clients of the control plane nodes after a successful deployment.

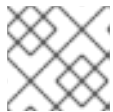


152_OpenShift_0421

OpenShift Container Platform nodes must agree on a date and time to run properly. When worker nodes retrieve the date and time from the NTP servers on the control plane nodes, it enables the installation and operation of clusters that are not connected to a routable network and thereby do not have access to a higher stratum NTP server.

Procedure

1. Create a Butane config, **99-master-chrony-conf-override.bu**, including the contents of the **chrony.conf** file for the control plane nodes.



NOTE

See "Creating machine configs with Butane" for information about Butane.

Butane config example

```
variant: openshift
version: 4.12.0
metadata:
  name: 99-master-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: master
storage:
files:
  - path: /etc/chrony.conf
    mode: 0644
    overwrite: true
    contents:
      inline: |
        # Use public servers from the pool.ntp.org project.
        # Please consider joining the pool (https://www.pool.ntp.org/join.html).

        # The Machine Config Operator manages this file
        server openshift-master-0.<cluster-name>.<domain> iburst 1
        server openshift-master-1.<cluster-name>.<domain> iburst
        server openshift-master-2.<cluster-name>.<domain> iburst

        stratumweight 0
```



```

driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony

# Configure the control plane nodes to serve as local NTP servers
# for all worker nodes, even if they are not in sync with an
# upstream NTP server.

# Allow NTP client access from the local network.
allow all
# Serve time even if not synchronized to a time source.
local stratum 3 orphan

```

- 1 You must replace **<cluster-name>** with the name of the cluster and replace **<domain>** with the fully qualified domain name.
2. Use Butane to generate a **MachineConfig** object file, **99-master-chrony-conf-override.yaml**, containing the configuration to be delivered to the control plane nodes:

```
$ butane 99-master-chrony-conf-override.bu -o 99-master-chrony-conf-override.yaml
```

3. Create a Butane config, **99-worker-chrony-conf-override.bu**, including the contents of the **chrony.conf** file for the worker nodes that references the NTP servers on the control plane nodes.

Butane config example

```

variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # The Machine Config Operator manages this file.
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0

```

```
driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony
```

- 1 You must replace **<cluster-name>** with the name of the cluster and replace **<domain>** with the fully qualified domain name.

4. Use Butane to generate a **MachineConfig** object file, **99-worker-chrony-conf-override.yaml**, containing the configuration to be delivered to the worker nodes:

```
$ butane 99-worker-chrony-conf-override.bu -o 99-worker-chrony-conf-override.yaml
```

5. Apply the **99-master-chrony-conf-override.yaml** policy to the control plane nodes.

```
$ oc apply -f 99-master-chrony-conf-override.yaml
```

Example output

```
machineconfig.machineconfiguration.openshift.io/99-master-chrony-conf-override created
```

6. Apply the **99-worker-chrony-conf-override.yaml** policy to the worker nodes.

```
$ oc apply -f 99-worker-chrony-conf-override.yaml
```

Example output

```
machineconfig.machineconfiguration.openshift.io/99-worker-chrony-conf-override created
```

7. Check the status of the applied NTP settings.

```
$ oc describe machineconfigpool
```

16.4.2. Enabling a provisioning network after installation

The assisted installer and installer-provisioned installation for bare metal clusters provide the ability to deploy a cluster without a **provisioning** network. This capability is for scenarios such as proof-of-concept clusters or deploying exclusively with Redfish virtual media when each node's baseboard management controller is routable via the **baremetal** network.

You can enable a **provisioning** network after installation using the Cluster Baremetal Operator (CBO).

Prerequisites

- A dedicated physical network must exist, connected to all worker and control plane nodes.
- You must isolate the native, untagged physical network.
- The network cannot have a DHCP server when the **provisioningNetwork** configuration setting is set to **Managed**.
- You can omit the **provisioningInterface** setting in OpenShift Container Platform 4.10 to use the **bootMACAddress** configuration setting.

Procedure

1. When setting the **provisioningInterface** setting, first identify the provisioning interface name for the cluster nodes. For example, **eth0** or **eno1**.
2. Enable the Preboot eXecution Environment (PXE) on the **provisioning** network interface of the cluster nodes.
3. Retrieve the current state of the **provisioning** network and save it to a provisioning custom resource (CR) file:

```
$ oc get provisioning -o yaml > enable-provisioning-nw.yaml
```

4. Modify the provisioning CR file:

```
$ vim ~/enable-provisioning-nw.yaml
```

Scroll down to the **provisioningNetwork** configuration setting and change it from **Disabled** to **Managed**. Then, add the **provisioningIP**, **provisioningNetworkCIDR**, **provisioningDHCPRange**, **provisioningInterface**, and **watchAllNameSpaces** configuration settings after the **provisioningNetwork** setting. Provide appropriate values for each setting.

```
apiVersion: v1
items:
- apiVersion: metal3.io/v1alpha1
  kind: Provisioning
  metadata:
    name: provisioning-configuration
  spec:
    provisioningNetwork: 1
    provisioningIP: 2
    provisioningNetworkCIDR: 3
    provisioningDHCPRange: 4
    provisioningInterface: 5
    watchAllNameSpaces: 6
```

- 1 The **provisioningNetwork** is one of **Managed**, **Unmanaged**, or **Disabled**. When set to **Managed**, Metal3 manages the provisioning network and the CBO deploys the Metal3 pod with a configured DHCP server. When set to **Unmanaged**, the system administrator configures the DHCP server manually.
- 2 The **provisioningIP** is the static IP address that the DHCP server and ironic use to provision the network. This static IP address must be within the **provisioning** subnet, and outside of the DHCP range. If you configure this setting, it must have a valid IP address

even if the **provisioning** network is **Disabled**. The static IP address is bound to the metal3 pod. If the metal3 pod fails and moves to another server, the static IP address also moves to the new server.

- 3 The Classless Inter-Domain Routing (CIDR) address. If you configure this setting, it must have a valid CIDR address even if the **provisioning** network is **Disabled**. For example: **192.168.0.1/24**.
- 4 The DHCP range. This setting is only applicable to a **Managed** provisioning network. Omit this configuration setting if the **provisioning** network is **Disabled**. For example: **192.168.0.64, 192.168.0.253**.
- 5 The NIC name for the **provisioning** interface on cluster nodes. The **provisioningInterface** setting is only applicable to **Managed** and **Unmanaged** provisioning networks. Omit the **provisioningInterface** configuration setting if the **provisioning** network is **Disabled**. Omit the **provisioningInterface** configuration setting to use the **bootMACAddress** configuration setting instead.
- 6 Set this setting to **true** if you want metal3 to watch namespaces other than the default **openshift-machine-api** namespace. The default value is **false**.

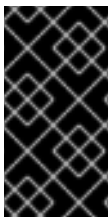
5. Save the changes to the provisioning CR file.

6. Apply the provisioning CR file to the cluster:

```
$ oc apply -f enable-provisioning-nw.yaml
```

16.4.3. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

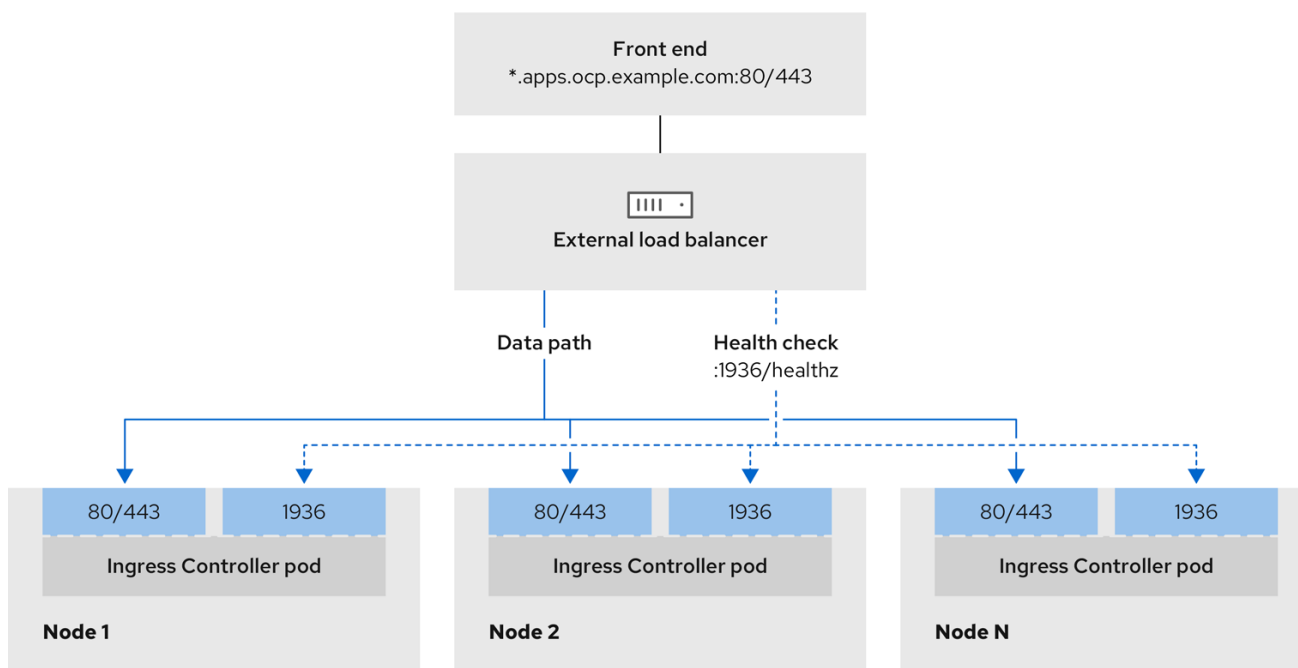
The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

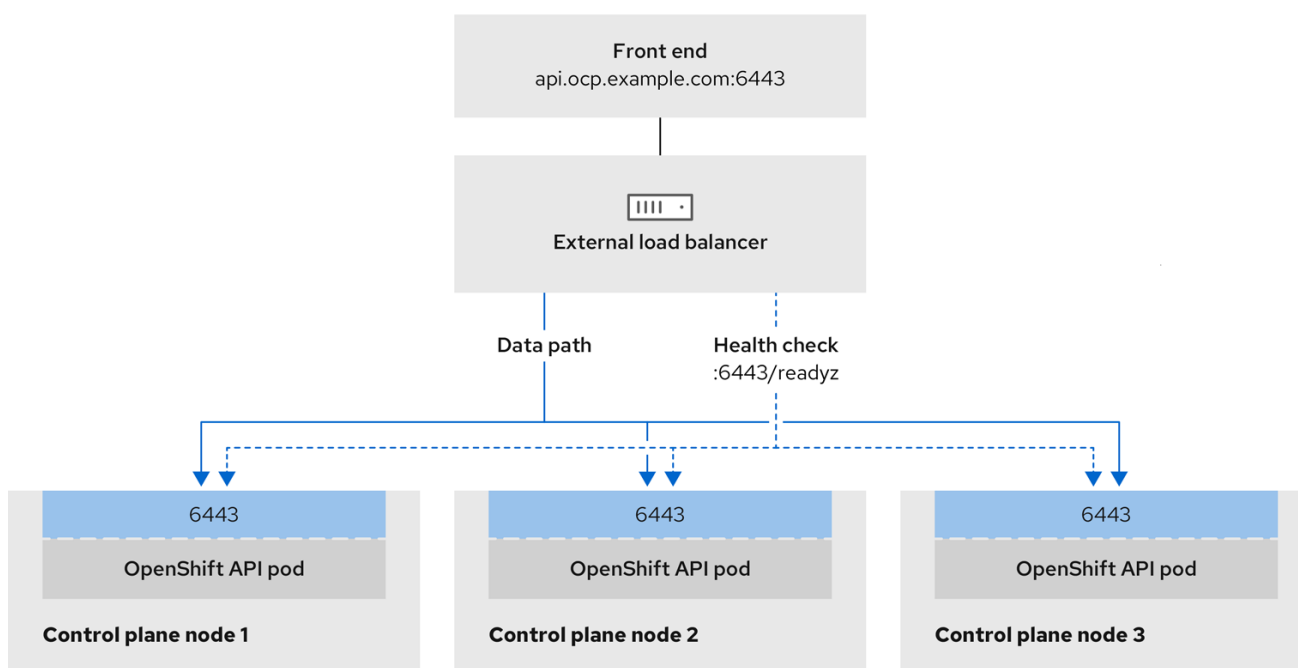
You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 16.1. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



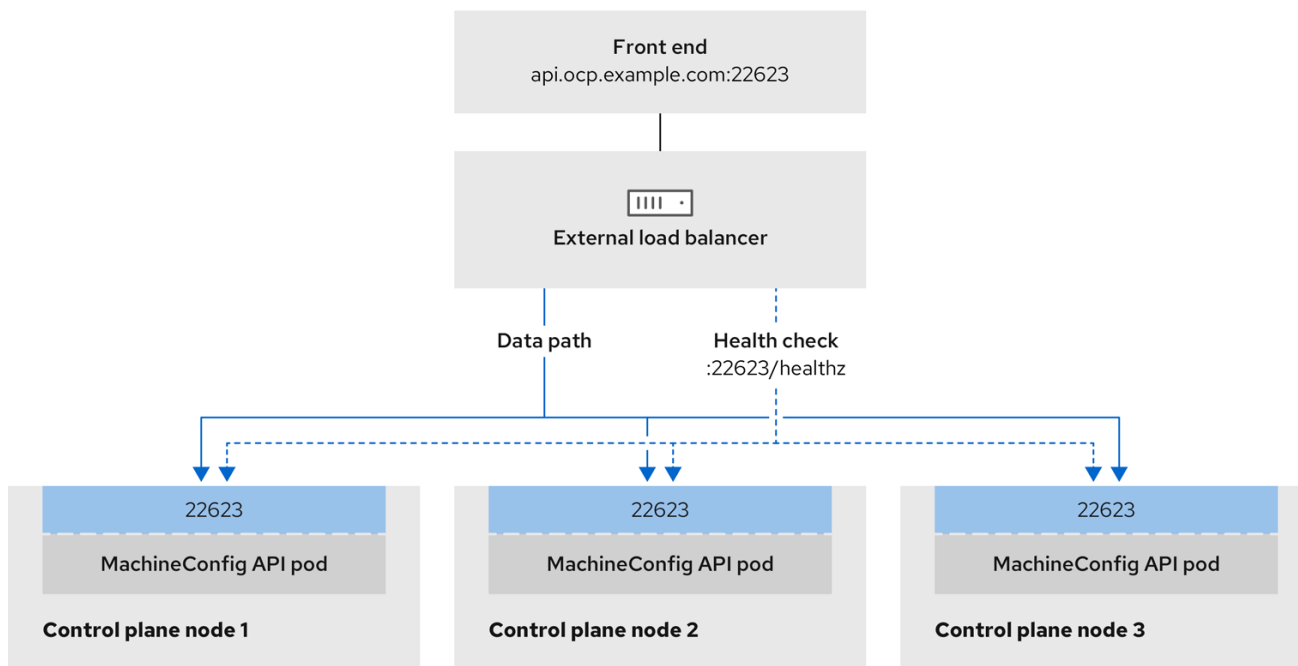
496_OpenShift_I223

Figure 16.2. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment



496_OpenShift_I223

Figure 16.3. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



496_OpenShift_1223

The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

16.4.3.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```



```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
}
```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private

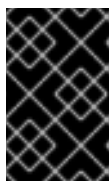
```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

16.5. EXPANDING THE CLUSTER

After deploying an installer-provisioned OpenShift Container Platform cluster, you can use the following procedures to expand the number of worker nodes. Ensure that each prospective worker node meets the prerequisites.

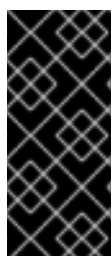


NOTE

Expanding the cluster using RedFish Virtual Media involves meeting minimum firmware requirements. See **Firmware requirements for installing with virtual media** in the **Prerequisites** section for additional details when expanding the cluster using RedFish Virtual Media.

16.5.1. Preparing the bare metal node

To expand your cluster, you must provide the node with the relevant IP address. This can be done with a static configuration, or with a DHCP (Dynamic Host Configuration protocol) server. When expanding the cluster using a DHCP server, each node must have a DHCP reservation.



RESERVING IP ADDRESSES SO THEY BECOME STATIC IP ADDRESSES

Some administrators prefer to use static IP addresses so that each node's IP address remains constant in the absence of a DHCP server. To configure static IP addresses with NMState, see "Optional: Configuring host network interfaces in the **install-config.yaml** file" in the "Setting up the environment for an OpenShift installation" section for additional details.

Preparing the bare metal node requires executing the following procedure from the provisioner node.

Procedure

1. Get the **oc** binary:

```
$ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/openshift-client-linux-$VERSION.tar.gz | tar zxvf - oc
```

```
$ sudo cp oc /usr/local/bin
```

2. Power off the bare metal node by using the baseboard management controller (BMC), and ensure it is off.
3. Retrieve the user name and password of the bare metal node's baseboard management controller. Then, create **base64** strings from the user name and password:

```
$ echo -ne "root" | base64
```

```
$ echo -ne "password" | base64
```

4. Create a configuration file for the bare metal node. Depending on whether you are using a static configuration or a DHCP server, use one of the following example **bmh.yaml** files, replacing values in the YAML to match your environment:

```
$ vim bmh.yaml
```

- Static configuration **bmh.yaml**:

```

---
apiVersion: v1 1
kind: Secret
metadata:
  name: openshift-worker-<num>-network-config-secret 2
  namespace: openshift-machine-api
type: Opaque
stringData:
  nmstate: | 3
  interfaces: 4
  - name: <nic1_name> 5
    type: ethernet
    state: up
    ipv4:
      address:
        - ip: <ip_address> 6
          prefix-length: 24
          enabled: true
    dns-resolver:
      config:
        server:
          - <dns_ip_address> 7
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: <next_hop_ip_address> 8
          next-hop-interface: <next_hop_nic1_name> 9
---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret 10
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid> 11
  password: <base64_of_pwd> 12
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num> 13
  namespace: openshift-machine-api
spec:
  online: True
  bootMACAddress: <nic1_mac_address> 14
  bmc:
    address: <protocol>://<bmc_url> 15
    credentialsName: openshift-worker-<num>-bmc-secret 16
    disableCertificateVerification: True 17
    username: <bmc_username> 18

```

```
password: <bmc_password> 19
rootDeviceHints:
  deviceName: <root_device_hint> 20
preprovisioningNetworkDataName: openshift-worker-<num>-network-config-secret 21
```

- 1 To configure the network interface for a newly created node, specify the name of the secret that contains the network configuration. Follow the **nmstate** syntax to define the network configuration for your node. See "Optional: Configuring host network interfaces in the install-config.yaml file" for details on configuring NMState syntax.
- 2 10 13 16 Replace **<num>** for the worker number of the bare metal node in the **name** fields, the **credentialsName** field, and the **preprovisioningNetworkDataName** field.
- 3 Add the NMState YAML syntax to configure the host interfaces.
- 4 Optional: If you have configured the network interface with **nmstate**, and you want to disable an interface, set **state: up** with the IP addresses set to **enabled: false** as shown:

```
---
interfaces:
- name: <nic_name>
  type: ethernet
  state: up
  ipv4:
    enabled: false
  ipv6:
    enabled: false
```

- 5 6 7 8 9 Replace **<nic1_name>**, **<ip_address>**, **<dns_ip_address>**, **<next_hop_ip_address>** and **<next_hop_nic1_name>** with appropriate values.
- 11 12 Replace **<base64_of_uid>** and **<base64_of_pwd>** with the base64 string of the user name and password.
- 14 Replace **<nic1_mac_address>** with the MAC address of the bare metal node's first NIC. See the "BMC addressing" section for additional BMC configuration options.
- 15 Replace **<protocol>** with the BMC protocol, such as IPMI, RedFish, or others. Replace **<bmc_url>** with the URL of the bare metal node's baseboard management controller.
- 17 To skip certificate validation, set **disableCertificateVerification** to true.
- 18 19 Replace **<bmc_username>** and **<bmc_password>** with the string of the BMC user name and password.
- 20 Optional: Replace **<root_device_hint>** with a device path if you specify a root device hint.
- 21 Optional: If you have configured the network interface for the newly created node, provide the network configuration secret name in the **preprovisioningNetworkDataName** of the BareMetalHost CR.

- DHCP configuration **bmh.yaml**:

```

---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret 1
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid> 2
  password: <base64_of_pwd> 3
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num> 4
  namespace: openshift-machine-api
spec:
  online: True
  bootMACAddress: <nic1_mac_address> 5
  bmc:
    address: <protocol>://<bmc_url> 6
    credentialsName: openshift-worker-<num>-bmc-secret 7
    disableCertificateVerification: True 8
    username: <bmc_username> 9
    password: <bmc_password> 10
  rootDeviceHints:
    deviceName: <root_device_hint> 11
  preprovisioningNetworkDataName: openshift-worker-<num>-network-config-secret 12

```

1 4 7 Replace **<num>** for the worker number of the bare metal node in the **name** fields, the **credentialsName** field, and the **preprovisioningNetworkDataName** field.

2 3 Replace **<base64_of_uid>** and **<base64_of_pwd>** with the base64 string of the user name and password.

5 Replace **<nic1_mac_address>** with the MAC address of the bare metal node's first NIC. See the "BMC addressing" section for additional BMC configuration options.

6 Replace **<protocol>** with the BMC protocol, such as IPMI, RedFish, or others. Replace **<bmc_url>** with the URL of the bare metal node's baseboard management controller.

8 To skip certificate validation, set **disableCertificateVerification** to true.

9 10 Replace **<bmc_username>** and **<bmc_password>** with the string of the BMC user name and password.

11 Optional: Replace **<root_device_hint>** with a device path if you specify a root device hint.

12 Optional: If you have configured the network interface for the newly created node, provide the network configuration secret name in the **preprovisioningNetworkDataName** of the BareMetalHost CR.

**NOTE**

If the MAC address of an existing bare metal node matches the MAC address of a bare metal host that you are attempting to provision, then the Ironi installation will fail. If the host enrollment, inspection, cleaning, or other Ironi steps fail, the Bare Metal Operator retries the installation continuously. See "Diagnosing a host duplicate MAC address" for more information.

5. Create the bare metal node:

```
$ oc -n openshift-machine-api create -f bmh.yaml
```

Example output

```
secret/openshift-worker-<num>-network-config-secret created
secret/openshift-worker-<num>-bmc-secret created
baremetalhost.metal3.io/openshift-worker-<num> created
```

Where **<num>** will be the worker number.

6. Power up and inspect the bare metal node:

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

Where **<num>** is the worker node number.

Example output

```
NAME                STATE    CONSUMER  ONLINE  ERROR
openshift-worker-<num> available      true
```

**NOTE**

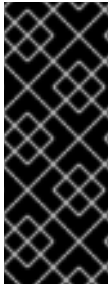
To allow the worker node to join the cluster, scale the **machineset** object to the number of the **BareMetalHost** objects. You can scale nodes either manually or automatically. To scale nodes automatically, use the **metal3.io/autoscale-to-hosts** annotation for **machineset**.

Additional resources

- See [Optional: Configuring host network interfaces in the install-config.yaml file](#) for details on configuring the NMState syntax.
- See [Automatically scaling machines to the number of available bare metal hosts](#) for details on automatically scaling machines.

16.5.2. Replacing a bare-metal control plane node

Use the following procedure to replace an installer-provisioned OpenShift Container Platform control plane node.



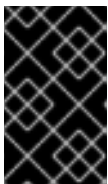
IMPORTANT

If you reuse the **BareMetalHost** object definition from an existing control plane host, do not leave the **externallyProvisioned** field set to **true**.

Existing control plane **BareMetalHost** objects may have the **externallyProvisioned** flag set to **true** if they were provisioned by the OpenShift Container Platform installation program.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have taken an etcd backup.



IMPORTANT

Take an etcd backup before performing this procedure so that you can restore your cluster if you encounter any issues. For more information about taking an etcd backup, see the *Additional resources* section.

Procedure

1. Ensure that the Bare Metal Operator is available:

```
$ oc get clusteroperator baremetal
```

Example output

```
NAME      VERSION AVAILABLE PROGRESSING DEGRADED SINCE MESSAGE
baremetal 4.12.0  True      False      False    3d15h
```

2. Remove the old **BareMetalHost** and **Machine** objects:

```
$ oc delete bmh -n openshift-machine-api <host_name>
$ oc delete machine -n openshift-machine-api <machine_name>
```

Replace **<host_name>** with the name of the host and **<machine_name>** with the name of the machine. The machine name appears under the **CONSUMER** field.

After you remove the **BareMetalHost** and **Machine** objects, then the machine controller automatically deletes the **Node** object.

3. Create the new **BareMetalHost** object and the secret to store the BMC credentials:

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: control-plane-<num>-bmc-secret 1
  namespace: openshift-machine-api
data:
  username: <base64_of_uid> 2
  password: <base64_of_pwd> 3
```

```

type: Opaque
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: control-plane-<num> 4
  namespace: openshift-machine-api
spec:
  automatedCleaningMode: disabled
  bmc:
    address: <protocol>://<bmc_ip> 5
    credentialsName: control-plane-<num>-bmc-secret 6
  bootMACAddress: <NIC1_mac_address> 7
  bootMode: UEFI
  externallyProvisioned: false
  online: true
EOF

```

- 1 4 6 Replace **<num>** for the control plane number of the bare metal node in the **name** fields and the **credentialsName** field.
- 2 Replace **<base64_of_uid>** with the **base64** string of the user name.
- 3 Replace **<base64_of_pwd>** with the **base64** string of the password.
- 5 Replace **<protocol>** with the BMC protocol, such as **redfish**, **redfish-virtualmedia**, **idrac-virtualmedia**, or others. Replace **<bmc_ip>** with the IP address of the bare metal node's baseboard management controller. For additional BMC configuration options, see "BMC addressing" in the *Additional resources* section.
- 7 Replace **<NIC1_mac_address>** with the MAC address of the bare metal node's first NIC.

After the inspection is complete, the **BareMetalHost** object is created and available to be provisioned.

4. View available **BareMetalHost** objects:

```
$ oc get bmh -n openshift-machine-api
```

Example output

NAME	STATE	CONSUMER	ONLINE	ERROR	AGE
control-plane-1.example.com	available	control-plane-1	true		1h10m
control-plane-2.example.com	externally provisioned	control-plane-2		true	4h53m
control-plane-3.example.com	externally provisioned	control-plane-3		true	4h53m
compute-1.example.com	provisioned	compute-1-ktmmx	true		4h53m
compute-1.example.com	provisioned	compute-2-l2zmb	true		4h53m

There are no **MachineSet** objects for control plane nodes, so you must create a **Machine** object instead. You can copy the **providerSpec** from another control plane **Machine** object.

5. Create a **Machine** object:

```

$ cat <<EOF | oc apply -f -
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  annotations:
    metal3.io/BareMetalHost: openshift-machine-api/control-plane-<num> 1
  labels:
    machine.openshift.io/cluster-api-cluster: control-plane-<num> 2
    machine.openshift.io/cluster-api-machine-role: master
    machine.openshift.io/cluster-api-machine-type: master
  name: control-plane-<num> 3
  namespace: openshift-machine-api
spec:
  metadata: {}
  providerSpec:
    value:
      apiVersion: baremetal.cluster.k8s.io/v1alpha1
      customDeploy:
        method: install_coreos
      hostSelector: {}
      image:
        checksum: ""
        url: ""
      kind: BareMetalMachineProviderSpec
      metadata:
        creationTimestamp: null
      userData:
        name: master-user-data-managed
EOF

```

1 2 3 Replace **<num>** for the control plane number of the bare metal node in the **name**, **labels** and **annotations** fields.

6. To view the **BareMetalHost** objects, run the following command:

```
$ oc get bmh -A
```

Example output

NAME	STATE	CONSUMER	ONLINE	ERROR	AGE
control-plane-1.example.com	provisioned	control-plane-1	true		2h53m
control-plane-2.example.com	externally provisioned	control-plane-2	true		5h53m
control-plane-3.example.com	externally provisioned	control-plane-3	true		5h53m
compute-1.example.com	provisioned	compute-1-ktmmx	true		5h53m
compute-2.example.com	provisioned	compute-2-l2zmb	true		5h53m

7. After the RHCOS installation, verify that the **BareMetalHost** is added to the cluster:

■

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
control-plane-1.example.com	available	master	4m2s	v1.18.2
control-plane-2.example.com	available	master	141m	v1.18.2
control-plane-3.example.com	available	master	141m	v1.18.2
compute-1.example.com	available	worker	87m	v1.18.2
compute-2.example.com	available	worker	87m	v1.18.2



NOTE

After replacement of the new control plane node, the etcd pod running in the new node is in **crashloopback** status. See "Replacing an unhealthy etcd member" in the *Additional resources* section for more information.

Additional resources

- [Replacing an unhealthy etcd member](#)
- [Backing up etcd](#)
- [Bare metal configuration](#)
- [BMC addressing](#)

16.5.3. Preparing to deploy with Virtual Media on the baremetal network

If the **provisioning** network is enabled and you want to expand the cluster using Virtual Media on the **baremetal** network, use the following procedure.

Prerequisites

- There is an existing cluster with a **baremetal** network and a **provisioning** network.

Procedure

1. Edit the **provisioning** custom resource (CR) to enable deploying with Virtual Media on the **baremetal** network:

```
oc edit provisioning

apiVersion: metal3.io/v1alpha1
kind: Provisioning
metadata:
  creationTimestamp: "2021-08-05T18:51:50Z"
  finalizers:
  - provisioning.metal3.io
  generation: 8
  name: provisioning-configuration
  resourceVersion: "551591"
  uid: f76e956f-24c6-4361-aa5b-feaf72c5b526
```

```

spec:
  provisioningDHCPRange: 172.22.0.10,172.22.0.254
  provisioningIP: 172.22.0.3
  provisioningInterface: enp1s0
  provisioningNetwork: Managed
  provisioningNetworkCIDR: 172.22.0.0/24
  virtualMediaViaExternalNetwork: true 1
status:
  generations:
    - group: apps
      hash: ""
      lastGeneration: 7
      name: metal3
      namespace: openshift-machine-api
      resource: deployments
    - group: apps
      hash: ""
      lastGeneration: 1
      name: metal3-image-cache
      namespace: openshift-machine-api
      resource: daemonsets
  observedGeneration: 8
  readyReplicas: 0

```

1 Add **virtualMediaViaExternalNetwork: true** to the **provisioning** CR.

- If the image URL exists, edit the **machineset** to use the API VIP address. This step only applies to clusters installed in versions 4.9 or earlier.

```
oc edit machineset
```

```

apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  creationTimestamp: "2021-08-05T18:51:52Z"
  generation: 11
  labels:
    machine.openshift.io/cluster-api-cluster: ostest-hwmdt
    machine.openshift.io/cluster-api-machine-role: worker
    machine.openshift.io/cluster-api-machine-type: worker
  name: ostest-hwmdt-worker-0
  namespace: openshift-machine-api
  resourceVersion: "551513"
  uid: fad1c6e0-b9da-4d4a-8d73-286f78788931
spec:
  replicas: 2
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: ostest-hwmdt
      machine.openshift.io/cluster-api-machineset: ostest-hwmdt-worker-0
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: ostest-hwmdt

```

```

machine.openshift.io/cluster-api-machine-role: worker
machine.openshift.io/cluster-api-machine-type: worker
machine.openshift.io/cluster-api-machineset: ostest-hwmdt-worker-0
spec:
  metadata: {}
  providerSpec:
    value:
      apiVersion: baremetal.cluster.k8s.io/v1alpha1
      hostSelector: {}
      image:
        checksum: http://172.22.0.3:6181/images/rhcos-<version>.<architecture>.qcow2.
<md5sum> 1
        url: http://172.22.0.3:6181/images/rhcos-<version>.<architecture>.qcow2 2
      kind: BareMetalMachineProviderSpec
      metadata:
        creationTimestamp: null
      userData:
        name: worker-user-data
  status:
    availableReplicas: 2
    fullyLabeledReplicas: 2
    observedGeneration: 11
    readyReplicas: 2
    replicas: 2

```

1 Edit the **checksum** URL to use the API VIP address.

2 Edit the **url** URL to use the API VIP address.

16.5.4. Diagnosing a duplicate MAC address when provisioning a new host in the cluster

If the MAC address of an existing bare-metal node in the cluster matches the MAC address of a bare-metal host you are attempting to add to the cluster, the Bare Metal Operator associates the host with the existing node. If the host enrollment, inspection, cleaning, or other Ironic steps fail, the Bare Metal Operator retries the installation continuously. A registration error is displayed for the failed bare-metal host.

You can diagnose a duplicate MAC address by examining the bare-metal hosts that are running in the **openshift-machine-api** namespace.

Prerequisites

- Install an OpenShift Container Platform cluster on bare metal.
- Install the OpenShift Container Platform CLI **oc**.
- Log in as a user with **cluster-admin** privileges.

Procedure

To determine whether a bare-metal host that fails provisioning has the same MAC address as an existing node, do the following:

1. Get the bare-metal hosts running in the **openshift-machine-api** namespace:

```
$ oc get bmh -n openshift-machine-api
```

Example output

```
NAME                STATUS  PROVISIONING STATUS  CONSUMER
openshift-master-0  OK     externally provisioned openshift-zpwpq-master-0
openshift-master-1  OK     externally provisioned openshift-zpwpq-master-1
openshift-master-2  OK     externally provisioned openshift-zpwpq-master-2
openshift-worker-0  OK     provisioned          openshift-zpwpq-worker-0-lv84n
openshift-worker-1  OK     provisioned          openshift-zpwpq-worker-0-zd8lm
openshift-worker-2  error  registering
```

- To see more detailed information about the status of the failing host, run the following command replacing **<bare_metal_host_name>** with the name of the host:

```
$ oc get -n openshift-machine-api bmh <bare_metal_host_name> -o yaml
```

Example output

```
...
status:
  errorCount: 12
  errorMessage: MAC address b4:96:91:1d:7c:20 conflicts with existing node openshift-
worker-1
  errorType: registration error
...
```

16.5.5. Provisioning the bare metal node

Provisioning the bare metal node requires executing the following procedure from the provisioner node.

Procedure

- Ensure the **STATE** is **available** before provisioning the bare metal node.

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

Where **<num>** is the worker node number.

```
NAME                STATE  ONLINE ERROR AGE
openshift-worker  available true    34h
```

- Get a count of the number of worker nodes.

```
$ oc get nodes
```

```
NAME                                STATUS  ROLES    AGE  VERSION
openshift-master-1.openshift.example.com  Ready  master   30h  v1.25.0
openshift-master-2.openshift.example.com  Ready  master   30h  v1.25.0
openshift-master-3.openshift.example.com  Ready  master   30h  v1.25.0
openshift-worker-0.openshift.example.com  Ready  worker   30h  v1.25.0
openshift-worker-1.openshift.example.com  Ready  worker   30h  v1.25.0
```


- Get the compute machine set.

```
$ oc get machinesets -n openshift-machine-api
```

NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
...					
openshift-worker-0.example.com	1	1	1	1	55m
openshift-worker-1.example.com	1	1	1	1	55m

- Increase the number of worker nodes by one.

```
$ oc scale --replicas=<num> machineset <machineset> -n openshift-machine-api
```

Replace **<num>** with the new number of worker nodes. Replace **<machineset>** with the name of the compute machine set from the previous step.

- Check the status of the bare metal node.

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

Where **<num>** is the worker node number. The STATE changes from **ready** to **provisioning**.

NAME	STATE	CONSUMER	ONLINE	ERROR
openshift-worker-<num>	provisioning	openshift-worker-<num>-65tjz		true

The **provisioning** status remains until the OpenShift Container Platform cluster provisions the node. This can take 30 minutes or more. After the node is provisioned, the state will change to **provisioned**.

NAME	STATE	CONSUMER	ONLINE	ERROR
openshift-worker-<num>	provisioned	openshift-worker-<num>-65tjz		true

- After provisioning completes, ensure the bare metal node is ready.

```
$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
openshift-master-1.openshift.example.com	Ready	master	30h	v1.25.0
openshift-master-2.openshift.example.com	Ready	master	30h	v1.25.0
openshift-master-3.openshift.example.com	Ready	master	30h	v1.25.0
openshift-worker-0.openshift.example.com	Ready	worker	30h	v1.25.0
openshift-worker-1.openshift.example.com	Ready	worker	30h	v1.25.0
openshift-worker-<num>.openshift.example.com	Ready	worker	3m27s	v1.25.0

You can also check the kubelet.

```
$ ssh openshift-worker-<num>
```

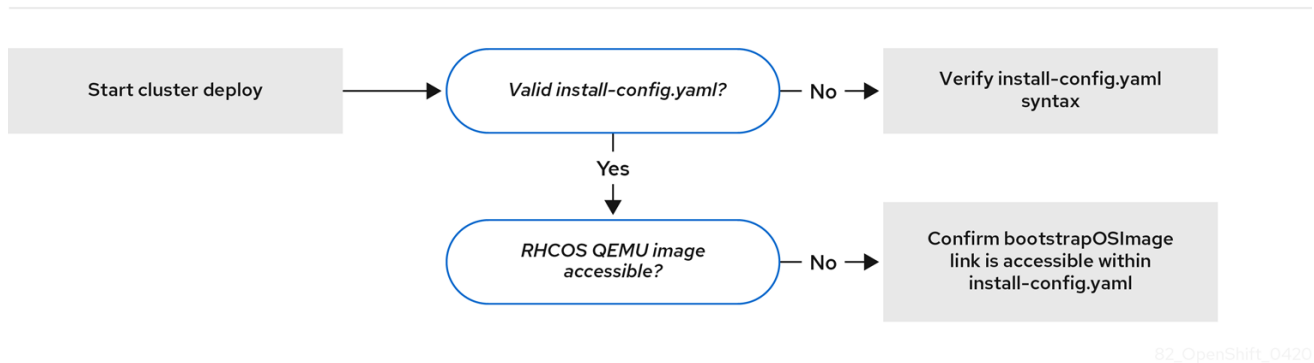
```
[kni@openshift-worker-<num>]$ journalctl -fu kubelet
```

16.6. TROUBLESHOOTING

16.6.1. Troubleshooting the installer workflow

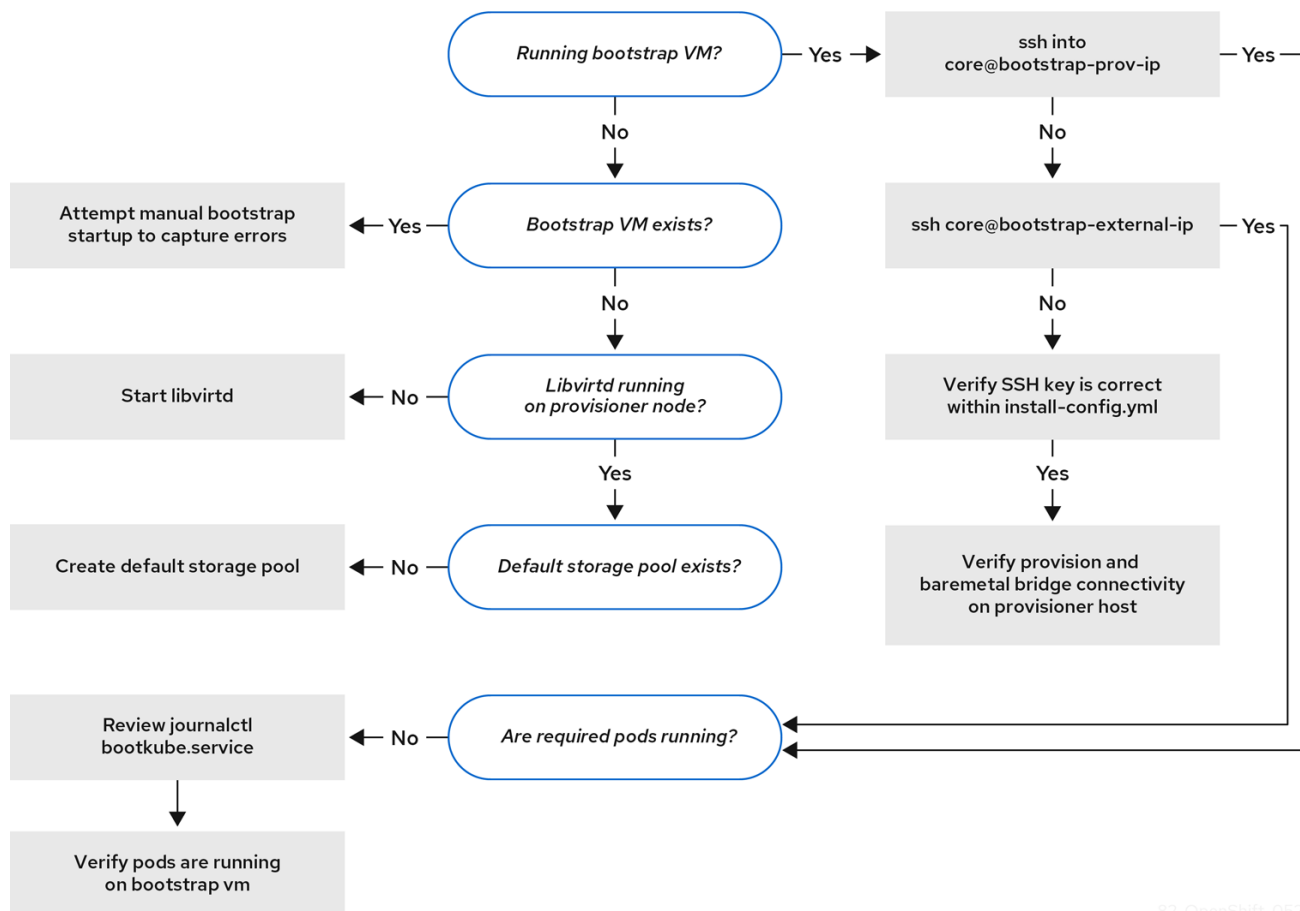
Prior to troubleshooting the installation environment, it is critical to understand the overall flow of the installer-provisioned installation on bare metal. The diagrams below provide a troubleshooting flow with a step-by-step breakdown for the environment.

Workflow 1 of 4



Workflow 1 of 4 illustrates a troubleshooting workflow when the **install-config.yaml** file has errors or the Red Hat Enterprise Linux CoreOS (RHCOS) images are inaccessible. Troubleshooting suggestions can be found at [Troubleshooting install-config.yaml](#).

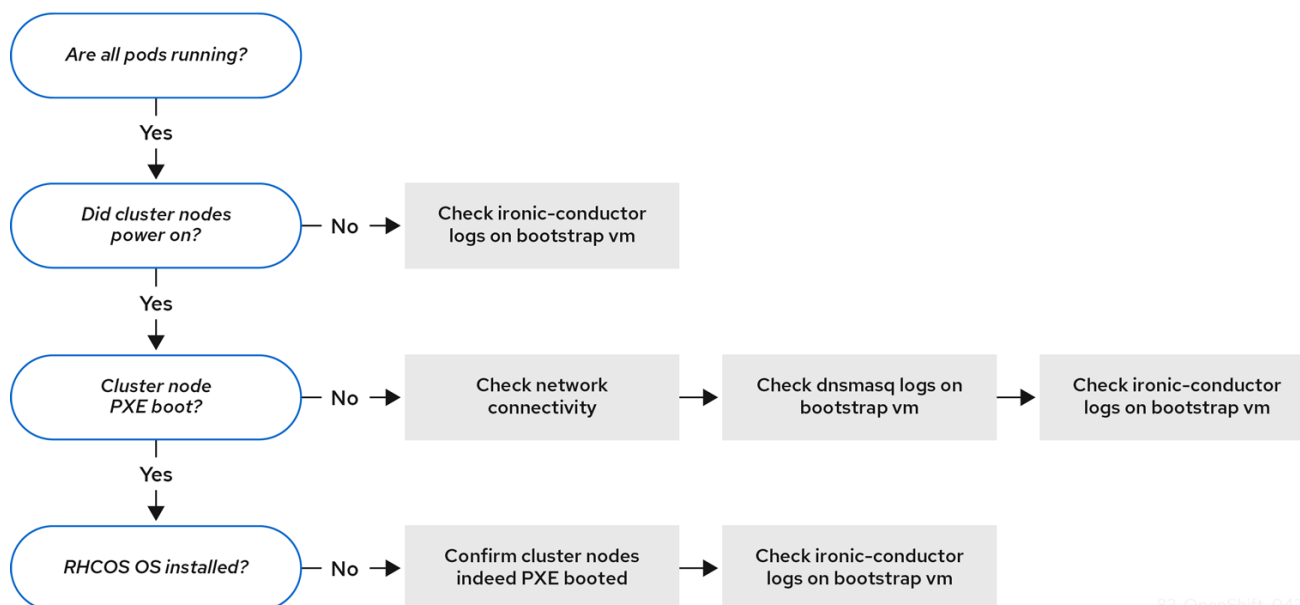
Workflow 2 of 4



82_OpenShift_0520

Workflow 2 of 4 illustrates a troubleshooting workflow for [bootstrap VM issues](#), [bootstrap VMs that cannot boot up the cluster nodes](#), and [inspecting logs](#). When installing an OpenShift Container Platform cluster without the **provisioning** network, this workflow does not apply.

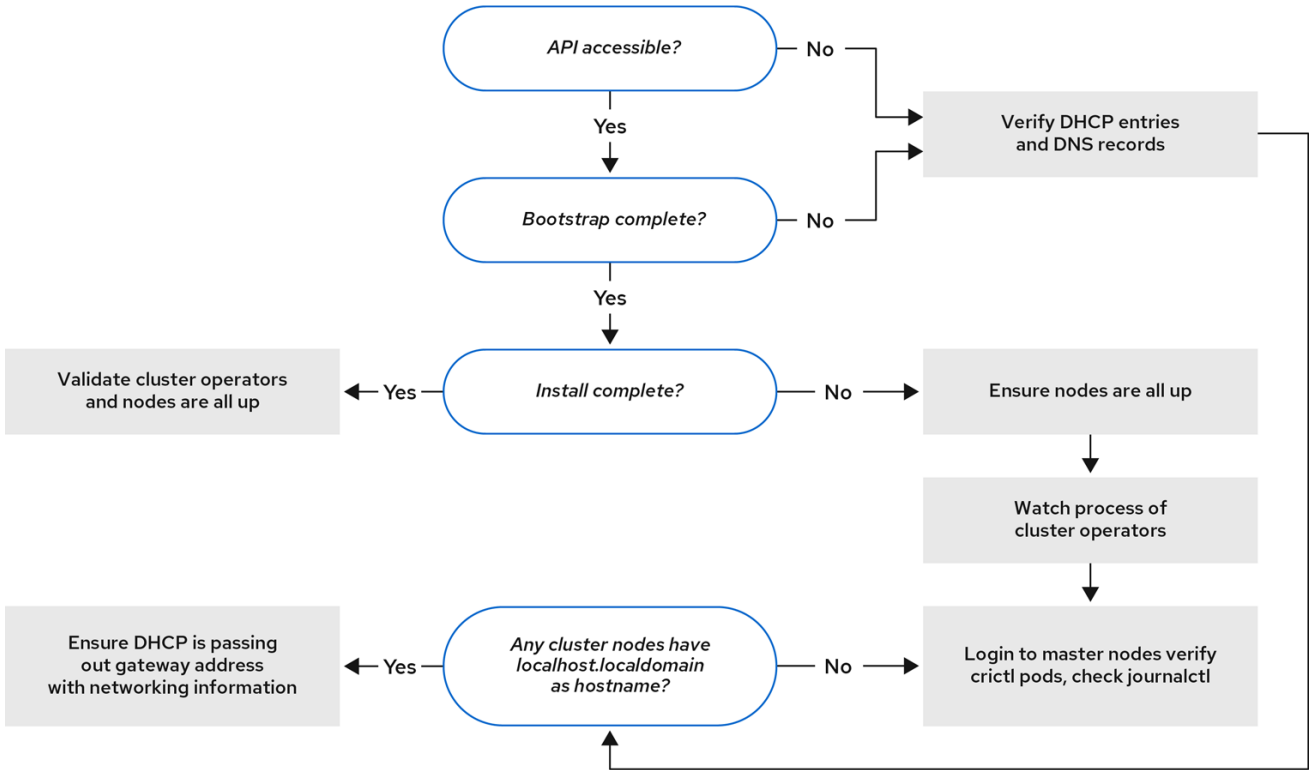
Workflow 3 of 4



82_OpenShift_0420

Workflow 3 of 4 illustrates a troubleshooting workflow for [cluster nodes that will not PXE boot](#). If installing using RedFish Virtual Media, each node must meet minimum firmware requirements for the installer to deploy the node. See **Firmware requirements for installing with virtual media** in the **Prerequisites** section for additional details.

Workflow 4 of 4



82_OpenShift_0420

Workflow 4 of 4 illustrates a troubleshooting workflow from [a non-accessible API](#) to a [validated installation](#).

16.6.2. Troubleshooting install-config.yaml

The **install-config.yaml** configuration file represents all of the nodes that are part of the OpenShift Container Platform cluster. The file contains the necessary options consisting of but not limited to **apiVersion**, **baseDomain**, **imageContentSources** and virtual IP addresses. If errors occur early in the deployment of the OpenShift Container Platform cluster, the errors are likely in the **install-config.yaml** configuration file.

Procedure

1. Use the guidelines in [YAML-tips](#).
2. Verify the YAML syntax is correct using [syntax-check](#).
3. Verify the Red Hat Enterprise Linux CoreOS (RHCOS) QEMU images are properly defined and accessible via the URL provided in the **install-config.yaml**. For example:

```
$ curl -s -o /dev/null -I -w "%{http_code}\n" http://webserver.example.com:8080/rhcos-44.81.202004250133-0-qemu.<architecture>.qcow2.gz?sha256=7d884b46ee54fe87bbc3893bf2aa99af3b2d31f2e19ab5529c60636fbd0f1ce7
```

If the output is **200**, there is a valid response from the webserver storing the bootstrap VM image.

16.6.3. Bootstrap VM issues

The OpenShift Container Platform installation program spawns a bootstrap node virtual machine, which handles provisioning the OpenShift Container Platform cluster nodes.

Procedure

1. About 10 to 15 minutes after triggering the installation program, check to ensure the bootstrap VM is operational using the **virsh** command:

```
$ sudo virsh list
```

Id	Name	State
12	openshift-xf6fq-bootstrap	running



NOTE

The name of the bootstrap VM is always the cluster name followed by a random set of characters and ending in the word "bootstrap."

If the bootstrap VM is not running after 10–15 minutes, troubleshoot why it is not running. Possible issues include:

2. Verify **libvirtd** is running on the system:

```
$ systemctl status libvirtd
```

```
● libvirtd.service - Virtualization daemon
  Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2020-03-03 21:21:07 UTC; 3 weeks 5 days ago
    Docs: man:libvirtd(8)
          https://libvirt.org
  Main PID: 9850 (libvirtd)
    Tasks: 20 (limit: 32768)
  Memory: 74.8M
  CGroup: /system.slice/libvirtd.service
          └─ 9850 /usr/sbin/libvirtd
```

If the bootstrap VM is operational, log in to it.

3. Use the **virsh console** command to find the IP address of the bootstrap VM:

```
$ sudo virsh console example.com
```

```

Connected to domain example.com
Escape character is ^]
Red Hat Enterprise Linux CoreOS 43.81.202001142154.0 (Ootpa) 4.3
SSH host key: SHA256:BRWJktXZgQQRY5zjuAV0IKZ4WM7i4TiUyMVanqu9Pqg (ED25519)
SSH host key: SHA256:7+iKGA7VtG5szmk2jB5gl/5EZ+SNcJ3a2g23o0Inlio (ECDSA)
SSH host key: SHA256:DH5VWhvhvagOTaLsYiVNse9ca+ZSW/30OOMed8rIGOc (RSA)
ens3: fd35:919d:4042:2:c7ed:9a9f:a9ec:7
ens4: 172.22.0.2 fe80::1d05:e52e:be5d:263f
localhost login:

```



IMPORTANT

When deploying an OpenShift Container Platform cluster without the **provisioning** network, you must use a public IP address and not a private IP address like **172.22.0.2**.

- After you obtain the IP address, log in to the bootstrap VM using the **ssh** command:



NOTE

In the console output of the previous step, you can use the IPv6 IP address provided by **ens3** or the IPv4 IP provided by **ens4**.

```
$ ssh core@172.22.0.2
```

If you are not successful logging in to the bootstrap VM, you have likely encountered one of the following scenarios:

- You cannot reach the **172.22.0.0/24** network. Verify the network connectivity between the provisioner and the **provisioning** network bridge. This issue might occur if you are using a **provisioning** network.
- You cannot reach the bootstrap VM through the public network. When attempting to SSH via **baremetal** network, verify connectivity on the **provisioner** host specifically around the **baremetal** network bridge.
- You encountered **Permission denied (publickey,password,keyboard-interactive)**. When attempting to access the bootstrap VM, a **Permission denied** error might occur. Verify that the SSH key for the user attempting to log in to the VM is set within the **install-config.yaml** file.

16.6.3.1. Bootstrap VM cannot boot up the cluster nodes

During the deployment, it is possible for the bootstrap VM to fail to boot the cluster nodes, which prevents the VM from provisioning the nodes with the RHCOS image. This scenario can arise due to:

- A problem with the **install-config.yaml** file.
- Issues with out-of-band network access when using the baremetal network.

To verify the issue, there are three containers related to **ironic**:

- ironic**
- ironic-inspector**

Procedure

1. Log in to the bootstrap VM:

```
$ ssh core@172.22.0.2
```

2. To check the container logs, execute the following:

```
[core@localhost ~]$ sudo podman logs -f <container_name>
```

Replace **<container_name>** with one of **ironic** or **ironic-inspector**. If you encounter an issue where the control plane nodes are not booting up from PXE, check the **ironic** pod. The **ironic** pod contains information about the attempt to boot the cluster nodes, because it attempts to log in to the node over IPMI.

Potential reason

The cluster nodes might be in the **ON** state when deployment started.

Solution

Power off the OpenShift Container Platform cluster nodes before you begin the installation over IPMI:

```
$ ipmitool -I lanplus -U root -P <password> -H <out_of_band_ip> power off
```

16.6.3.2. Inspecting logs

When experiencing issues downloading or accessing the RHCOS images, first verify that the URL is correct in the **install-config.yaml** configuration file.

Example of internal webserver hosting RHCOS images

```
bootstrapOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-qemu.<architecture>.qcow2.gz?
sha256=9d999f55ff1d44f7ed7c106508e5deecd04dc3c06095d34d36bf1cd127837e0c
clusterOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-openstack.<architecture>.qcow2.gz?
sha256=a1bda656fa0892f7b936fdc6b6a6086bddaed5dafacedcd7a1e811abb78fe3b0
```

The **coreos-downloader** container downloads resources from a webserver or from the external quay.io registry, whichever the **install-config.yaml** configuration file specifies. Verify that the **coreos-downloader** container is up and running and inspect its logs as needed.

Procedure

1. Log in to the bootstrap VM:

```
$ ssh core@172.22.0.2
```

2. Check the status of the **coreos-downloader** container within the bootstrap VM by running the following command:

```
[core@localhost ~]$ sudo podman logs -f coreos-downloader
```

If the bootstrap VM cannot access the URL to the images, use the **curl** command to verify that the VM can access the images.

- To inspect the **bootkube** logs that indicate if all the containers launched during the deployment phase, execute the following:

```
[core@localhost ~]$ journalctl -xe
```

```
[core@localhost ~]$ journalctl -b -f -u bootkube.service
```

- Verify all the pods, including **dnsmasq**, **mariadb**, **httpd**, and **ironic**, are running:

```
[core@localhost ~]$ sudo podman ps
```

- If there are issues with the pods, check the logs of the containers with issues. To check the logs of the **ironic** service, run the following command:

```
[core@localhost ~]$ sudo podman logs ironic
```

16.6.4. Cluster nodes will not PXE boot

When OpenShift Container Platform cluster nodes will not PXE boot, execute the following checks on the cluster nodes that will not PXE boot. This procedure does not apply when installing an OpenShift Container Platform cluster without the **provisioning** network.

Procedure

- Check the network connectivity to the **provisioning** network.
- Ensure PXE is enabled on the NIC for the **provisioning** network and PXE is disabled for all other NICs.
- Verify that the **install-config.yaml** configuration file includes the **rootDeviceHints** parameter and boot MAC address for the NIC connected to the **provisioning** network. For example:

control plane node settings

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
```

Worker node settings

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
```

16.6.5. Unable to discover new bare metal hosts using the BMC

In some cases, the installation program will not be able to discover the new bare metal hosts and issue an error, because it cannot mount the remote virtual media share.

For example:

```
ProvisioningError 51s metal3-baremetal-controller Image provisioning failed: Deploy step
deploy.deploy failed with BadRequestError: HTTP POST
https://<bmc_address>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia/CD/Actions/VirtualMedia.
InsertMedia
returned code 400.
```


Base.1.8.GeneralError: A general error has occurred. See ExtendedInfo for more information

```
Extended information: [
  {
    "Message": "Unable to mount remote share https://<ironic_address>/redfish/boot-<uuid>.iso.",
    "MessageArgs": [
      "https://<ironic_address>/redfish/boot-<uuid>.iso"
    ],
    "MessageArgs@odata.count": 1,
    "MessageId": "IDRAC.2.5.RAC0720",
    "RelatedProperties": [
      "#/Image"
    ],
    "RelatedProperties@odata.count": 1,
    "Resolution": "Retry the operation.",
    "Severity": "Informational"
  }
].
```

In this situation, if you are using virtual media with an unknown certificate authority, you can configure your baseboard management controller (BMC) remote file share settings to trust an unknown certificate authority to avoid this error.



NOTE

This resolution was tested on OpenShift Container Platform 4.11 with Dell iDRAC 9 and firmware version 5.10.50.

16.6.6. The API is not accessible

When the cluster is running and clients cannot access the API, domain name resolution issues might impede access to the API.

Procedure

1. **Hostname Resolution:** Check the cluster nodes to ensure they have a fully qualified domain name, and not just **localhost.localdomain**. For example:

```
$ hostname
```

If a hostname is not set, set the correct hostname. For example:

```
$ hostnamectl set-hostname <hostname>
```

2. **Incorrect Name Resolution:** Ensure that each node has the correct name resolution in the DNS server using **dig** and **nslookup**. For example:

```
$ dig api.<cluster_name>.example.com
```

```
; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el8 <<>> api.<cluster_name>.example.com
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 37551
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2
```

```

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; COOKIE: 866929d2f8e8563582af23f05ec44203d313e50948d43f60 (good)
;; QUESTION SECTION:
;api.<cluster_name>.example.com. IN A

;; ANSWER SECTION:
api.<cluster_name>.example.com. 10800 IN A 10.19.13.86

;; AUTHORITY SECTION:
<cluster_name>.example.com. 10800 IN NS <cluster_name>.example.com.

;; ADDITIONAL SECTION:
<cluster_name>.example.com. 10800 IN A 10.19.14.247

;; Query time: 0 msec
;; SERVER: 10.19.14.247#53(10.19.14.247)
;; WHEN: Tue May 19 20:30:59 UTC 2020
;; MSG SIZE rcvd: 140

```

The output in the foregoing example indicates that the appropriate IP address for the **api.<cluster_name>.example.com** VIP is **10.19.13.86**. This IP address should reside on the **baremetal** network.

16.6.7. Troubleshooting worker nodes that cannot join the cluster

Installer-provisioned clusters deploy with a DNS server that includes a DNS entry for the **api-int.<cluster_name>.<base_domain>** URL. If the nodes within the cluster use an external or upstream DNS server to resolve the **api-int.<cluster_name>.<base_domain>** URL and there is no such entry, worker nodes might fail to join the cluster. Ensure that all nodes in the cluster can resolve the domain name.

Procedure

1. Add a DNS A/AAAA or CNAME record to internally identify the API load balancer. For example, when using `dnsmasq`, modify the **dnsmasq.conf** configuration file:

```
$ sudo nano /etc/dnsmasq.conf
```

```

address=/api-int.<cluster_name>.<base_domain>/<IP_address>
address=/api-int.mycluster.example.com/192.168.1.10
address=/api-int.mycluster.example.com/2001:0db8:85a3:0000:0000:8a2e:0370:7334

```

2. Add a DNS PTR record to internally identify the API load balancer. For example, when using `dnsmasq`, modify the **dnsmasq.conf** configuration file:

```
$ sudo nano /etc/dnsmasq.conf
```

```

ptr-record=<IP_address>.in-addr.arpa,api-int.<cluster_name>.<base_domain>
ptr-record=10.1.168.192.in-addr.arpa,api-int.mycluster.example.com

```

3. Restart the DNS server. For example, when using `dnsmasq`, execute the following command:

```
$ sudo systemctl restart dnsmasq
```

These records must be resolvable from all the nodes within the cluster.

16.6.8. Cleaning up previous installations

In the event of a previous failed deployment, remove the artifacts from the failed attempt before attempting to deploy OpenShift Container Platform again.

Procedure

1. Power off all bare metal nodes prior to installing the OpenShift Container Platform cluster:

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management_server_ip> power off
```

2. Remove all old bootstrap resources if any are left over from a previous deployment attempt:

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

3. Remove the following from the **clusterconfigs** directory to prevent Terraform from failing:

```
$ rm -rf ~/clusterconfigs/auth ~/clusterconfigs/terraform* ~/clusterconfigs/tls
~/clusterconfigs/metadata.json
```

16.6.9. Issues with creating the registry

When creating a disconnected registry, you might encounter a "User Not Authorized" error when attempting to mirror the registry. This error might occur if you fail to append the new authentication to the existing **pull-secret.txt** file.

Procedure

1. Check to ensure authentication is successful:

```
$ /usr/local/bin/oc adm release mirror \
-a pull-secret-update.json \
--from=$UPSTREAM_REPO \
--to-release-image=$LOCAL_REG/$LOCAL_REPO:${VERSION} \
--to=$LOCAL_REG/$LOCAL_REPO
```

**NOTE**

Example output of the variables used to mirror the install images:

```
UPSTREAM_REPO=${RELEASE_IMAGE}
LOCAL_REG=<registry_FQDN>:<registry_port>
LOCAL_REPO='ocp4/openshift4'
```

The values of **RELEASE_IMAGE** and **VERSION** were set during the **Retrieving OpenShift Installer** step of the **Setting up the environment for an OpenShift installation** section.

2. After mirroring the registry, confirm that you can access it in your disconnected environment:

```
$ curl -k -u <user>:<password> https://registry.example.com:<registry_port>/v2/_catalog
{"repositories":["<Repo_Name>"]}
```

16.6.10. Miscellaneous issues

16.6.10.1. Addressing the runtime network not ready error

After the deployment of a cluster you might receive the following error:

```
`runtime network not ready: NetworkReady=false reason:NetworkPluginNotReady message:Network
plugin returns error: Missing CNI default network`
```

The Cluster Network Operator is responsible for deploying the networking components in response to a special object created by the installer. It runs very early in the installation process, after the control plane (master) nodes have come up, but before the bootstrap control plane has been torn down. It can be indicative of more subtle installer issues, such as long delays in bringing up control plane (master) nodes or issues with **apiserver** communication.

Procedure

1. Inspect the pods in the **openshift-network-operator** namespace:

```
$ oc get all -n openshift-network-operator
```

```
NAME                                READY STATUS      RESTARTS  AGE
pod/network-operator-69dfd7b577-bg89v  0/1  ContainerCreating  0        149m
```

2. On the **provisioner** node, determine that the network configuration exists:

```
$ kubectl get network.config.openshift.io cluster -oyaml
```

```
apiVersion: config.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  serviceNetwork:
  - 172.30.0.0/16
```

```
clusterNetwork:
- cidr: 10.128.0.0/14
  hostPrefix: 23
networkType: OVNKubernetes
```

If it does not exist, the installer did not create it. To determine why the installer did not create it, execute the following:

```
$ openshift-install create manifests
```

3. Check that the **network-operator** is running:

```
$ kubectl -n openshift-network-operator get pods
```

4. Retrieve the logs:

```
$ kubectl -n openshift-network-operator logs -l "name=network-operator"
```

On high availability clusters with three or more control plane (master) nodes, the Operator will perform leader election and all other Operators will sleep. For additional details, see [Troubleshooting](#).

16.6.10.2. Addressing the "No disk found with matching rootDeviceHints" error message

After you deploy a cluster, you might receive the following error message:

```
No disk found with matching rootDeviceHints
```

To address the **No disk found with matching rootDeviceHints** error message, a temporary workaround is to change the **rootDeviceHints** to **minSizeGigabytes: 300**.

After you change the **rootDeviceHints** settings, boot the CoreOS and then verify the disk information by using the following command:

```
$ udevadm info /dev/sda
```

If you are using DL360 Gen 10 servers, be aware that they have an SD-card slot that might be assigned the **/dev/sda** device name. If no SD card is present in the server, it can cause conflicts. Ensure that the SD card slot is disabled in the server's BIOS settings.

If the **minSizeGigabytes** workaround is not fulfilling the requirements, you might need to revert **rootDeviceHints** back to **/dev/sda**. This change allows ironic images to boot successfully.

An alternative approach to fixing this problem is by using the serial ID of the disk. However, be aware that finding the serial ID can be challenging and might make the configuration file less readable. If you choose this path, ensure that you gather the serial ID using the previously documented command and incorporate it into your configuration.

16.6.10.3. Cluster nodes not getting the correct IPv6 address over DHCP

If the cluster nodes are not getting the correct IPv6 address over DHCP, check the following:

1. Ensure the reserved IPv6 addresses reside outside the DHCP range.

2. In the IP address reservation on the DHCP server, ensure the reservation specifies the correct DHCP Unique Identifier (DUID). For example:

```
# This is a dnsmasq dhcp reservation, 'id:00:03:00:01' is the client id and '18:db:f2:8c:d5:9f' is
the MAC Address for the NIC
id:00:03:00:01:18:db:f2:8c:d5:9f,openshift-master-1,[2620:52:0:1302::6]
```

3. Ensure that route announcements are working.
4. Ensure that the DHCP server is listening on the required interfaces serving the IP address ranges.

16.6.10.4. Cluster nodes not getting the correct hostname over DHCP

During IPv6 deployment, cluster nodes must get their hostname over DHCP. Sometimes the **NetworkManager** does not assign the hostname immediately. A control plane (master) node might report an error such as:

```
Failed Units: 2
NetworkManager-wait-online.service
nodeip-configuration.service
```

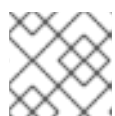
This error indicates that the cluster node likely booted without first receiving a hostname from the DHCP server, which causes **kubelet** to boot with a **localhost.localdomain** hostname. To address the error, force the node to renew the hostname.

Procedure

1. Retrieve the **hostname**:

```
[core@master-X ~]$ hostname
```

If the hostname is **localhost**, proceed with the following steps.



NOTE

Where **X** is the control plane node number.

2. Force the cluster node to renew the DHCP lease:

```
[core@master-X ~]$ sudo nmcli con up "<bare_metal_nic>"
```

Replace **<bare_metal_nic>** with the wired connection corresponding to the **baremetal** network.

3. Check **hostname** again:

```
[core@master-X ~]$ hostname
```

4. If the hostname is still **localhost.localdomain**, restart **NetworkManager**:

```
[core@master-X ~]$ sudo systemctl restart NetworkManager
```

- If the hostname is still **localhost.localdomain**, wait a few minutes and check again. If the hostname remains **localhost.localdomain**, repeat the previous steps.
- Restart the **nodeip-configuration** service:

```
[core@master-X ~]$ sudo systemctl restart nodeip-configuration.service
```

This service will reconfigure the **kubelet** service with the correct hostname references.

- Reload the unit files definition since the kubelet changed in the previous step:

```
[core@master-X ~]$ sudo systemctl daemon-reload
```

- Restart the **kubelet** service:

```
[core@master-X ~]$ sudo systemctl restart kubelet.service
```

- Ensure **kubelet** booted with the correct hostname:

```
[core@master-X ~]$ sudo journalctl -fu kubelet.service
```

If the cluster node is not getting the correct hostname over DHCP after the cluster is up and running, such as during a reboot, the cluster will have a pending **csr**. **Do not** approve a **csr**, or other issues might arise.

Addressing a csr

- Get CSRs on the cluster:

```
$ oc get csr
```

- Verify if a pending **csr** contains **Subject Name: localhost.localdomain**:

```
$ oc get csr <pending_csr> -o jsonpath='{.spec.request}' | base64 --decode | openssl req -noout -text
```

- Remove any **csr** that contains **Subject Name: localhost.localdomain**:

```
$ oc delete csr <wrong_csr>
```

16.6.10.5. Routes do not reach endpoints

During the installation process, it is possible to encounter a Virtual Router Redundancy Protocol (VRRP) conflict. This conflict might occur if a previously used OpenShift Container Platform node that was once part of a cluster deployment using a specific cluster name is still running but not part of the current OpenShift Container Platform cluster deployment using that same cluster name. For example, a cluster was deployed using the cluster name **openshift**, deploying three control plane (master) nodes and three worker nodes. Later, a separate install uses the same cluster name **openshift**, but this redeployment only installed three control plane (master) nodes, leaving the three worker nodes from a previous deployment in an **ON** state. This might cause a Virtual Router Identifier (VRID) conflict and a VRRP conflict.

- Get the route:

```
$ oc get route oauth-openshift
```

2. Check the service endpoint:

```
$ oc get svc oauth-openshift
```

```
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP  PORT(S)  AGE
oauth-openshift ClusterIP    172.30.19.162 <none>      443/TCP  59m
```

3. Attempt to reach the service from a control plane (master) node:

```
[core@master0 ~]$ curl -k https://172.30.19.162
```

```
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {
  },
  "status": "Failure",
  "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
  "reason": "Forbidden",
  "details": {
  },
  "code": 403
}
```

4. Identify the **authentication-operator** errors from the **provisioner** node:

```
$ oc logs deployment/authentication-operator -n openshift-authentication-operator
```

```
Event(v1.ObjectReference{Kind:"Deployment", Namespace:"openshift-authentication-operator", Name:"authentication-operator", UID:"225c5bd5-b368-439b-9155-5fd3c0459d98", APIVersion:"apps/v1", ResourceVersion:"", FieldPath:""}): type: 'Normal' reason: 'OperatorStatusChanged' Status for clusteroperator/authentication changed: Degraded message changed from "IngressStateEndpointsDegraded: All 2 endpoints for oauth-server are reporting"
```

Solution

1. Ensure that the cluster name for every deployment is unique, ensuring no conflict.
2. Turn off all the rogue nodes which are not part of the cluster deployment that are using the same cluster name. Otherwise, the authentication pod of the OpenShift Container Platform cluster might never start successfully.

16.6.10.6. Failed Ignition during Firstboot

During the Firstboot, the Ignition configuration may fail.

Procedure

1. Connect to the node where the Ignition configuration failed:


```
Failed Units: 1
machine-config-daemon-firstboot.service
```

- Restart the **machine-config-daemon-firstboot** service:

```
[core@worker-X ~]$ sudo systemctl restart machine-config-daemon-firstboot.service
```

16.6.10.7. NTP out of sync

The deployment of OpenShift Container Platform clusters depends on NTP synchronized clocks among the cluster nodes. Without synchronized clocks, the deployment may fail due to clock drift if the time difference is greater than two seconds.

Procedure

- Check for differences in the **AGE** of the cluster nodes. For example:

```
$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
master-0.cloud.example.com	Ready	master	145m	v1.25.0
master-1.cloud.example.com	Ready	master	135m	v1.25.0
master-2.cloud.example.com	Ready	master	145m	v1.25.0
worker-2.cloud.example.com	Ready	worker	100m	v1.25.0

- Check for inconsistent timing delays due to clock drift. For example:

```
$ oc get bmh -n openshift-machine-api
```

```
master-1 error registering master-1 ipmi://<out_of_band_ip>
```

```
$ sudo timedatectl
```

```
Local time: Tue 2020-03-10 18:20:02 UTC
Universal time: Tue 2020-03-10 18:20:02 UTC
RTC time: Tue 2020-03-10 18:36:53
Time zone: UTC (UTC, +0000)
System clock synchronized: no
NTP service: active
RTC in local TZ: no
```

Addressing clock drift in existing clusters

- Create a Butane config file including the contents of the **chrony.conf** file to be delivered to the nodes. In the following example, create **99-master-chrony.bu** to add the file to the control plane nodes. You can modify the file for worker nodes or repeat this procedure for the worker role.



NOTE

See "Creating machine configs with Butane" for information about Butane.

```

variant: openshift
version: 4.12.0
metadata:
  name: 99-master-chrony
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644
    overwrite: true
    contents:
      inline: |
        server <NTP_server> iburst 1
        stratumweight 0
        driftfile /var/lib/chrony/drift
        rtcsync
        makestep 10 3
        bindcmdaddress 127.0.0.1
        bindcmdaddress ::1
        keyfile /etc/chrony.keys
        commandkey 1
        generatecommandkey
        noclientlog
        logchange 0.5
        logdir /var/log/chrony

```

1 Replace **<NTP_server>** with the IP address of the NTP server.

2. Use Butane to generate a **MachineConfig** object file, **99-master-chrony.yaml**, containing the configuration to be delivered to the nodes:

```
$ butane 99-master-chrony.bu -o 99-master-chrony.yaml
```

3. Apply the **MachineConfig** object file:

```
$ oc apply -f 99-master-chrony.yaml
```

4. Ensure the **System clock synchronized** value is **yes**:

```
$ sudo timedatectl
```

```

Local time: Tue 2020-03-10 19:10:02 UTC
Universal time: Tue 2020-03-10 19:10:02 UTC
RTC time: Tue 2020-03-10 19:36:53
Time zone: UTC (UTC, +0000)
System clock synchronized: yes
NTP service: active
RTC in local TZ: no

```

To setup clock synchronization prior to deployment, generate the manifest files and add this file to the **openshift** directory. For example:

```
$ cp chrony-masters.yaml ~/clusterconfigs/openshift/99_masters-chrony-configuration.yaml
```

Then, continue to create the cluster.

16.6.11. Reviewing the installation

After installation, ensure the installer deployed the nodes and pods successfully.

Procedure

1. When the OpenShift Container Platform cluster nodes are installed appropriately, the following **Ready** state is seen within the **STATUS** column:

```
$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
master-0.example.com	Ready	master,worker	4h	v1.25.0
master-1.example.com	Ready	master,worker	4h	v1.25.0
master-2.example.com	Ready	master,worker	4h	v1.25.0

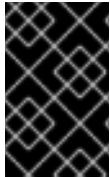
2. Confirm the installer deployed all pods successfully. The following command removes any pods that are still running or have completed as part of the output.

```
$ oc get pods --all-namespaces | grep -iv running | grep -iv complete
```

CHAPTER 17. INSTALLING IBM CLOUD BARE METAL (CLASSIC)

17.1. PREREQUISITES

You can use installer-provisioned installation to install OpenShift Container Platform on IBM Cloud® Bare Metal (Classic) nodes. This document describes the prerequisites and procedures when installing OpenShift Container Platform on IBM Cloud nodes.



IMPORTANT

Red Hat supports IPMI and PXE on the provisioning network only. Red Hat has not tested Red Fish, virtual media, or other complementary technologies such as Secure Boot on IBM Cloud deployments. A provisioning network is required.

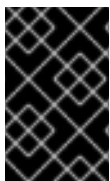
Installer-provisioned installation of OpenShift Container Platform requires:

- One node with Red Hat Enterprise Linux CoreOS (RHCOS) 8.x installed, for running the provisioner
- Three control plane nodes
- One routable network
- One provisioning network

Before starting an installer-provisioned installation of OpenShift Container Platform on IBM Cloud Bare Metal (Classic), address the following prerequisites and requirements.

17.1.1. Setting up IBM Cloud Bare Metal (Classic) infrastructure

To deploy an OpenShift Container Platform cluster on IBM Cloud® Bare Metal (Classic) infrastructure, you must first provision the IBM Cloud nodes.



IMPORTANT

Red Hat supports IPMI and PXE on the **provisioning** network only. Red Hat has not tested Red Fish, virtual media, or other complementary technologies such as Secure Boot on IBM Cloud deployments. The **provisioning** network is required.

You can customize IBM Cloud nodes using the IBM Cloud API. When creating IBM Cloud nodes, you must consider the following requirements.

Use one data center per cluster

All nodes in the OpenShift Container Platform cluster must run in the same IBM Cloud data center.

Create public and private VLANs

Create all nodes with a single public VLAN and a single private VLAN.

Ensure subnets have sufficient IP addresses

IBM Cloud public VLAN subnets use a /28 prefix by default, which provides 16 IP addresses. That is sufficient for a cluster consisting of three control plane nodes, four worker nodes, and two IP addresses for the API VIP and Ingress VIP on the **baremetal** network. For larger clusters, you might need a smaller

prefix.

IBM Cloud private VLAN subnets use a **/26** prefix by default, which provides 64 IP addresses. IBM Cloud Bare Metal (Classic) uses private network IP addresses to access the Baseboard Management Controller (BMC) of each node. OpenShift Container Platform creates an additional subnet for the **provisioning** network. Network traffic for the **provisioning** network subnet routes through the private VLAN. For larger clusters, you might need a smaller prefix.

Table 17.1. IP addresses per prefix

IP addresses	Prefix
32	/27
64	/26
128	/25
256	/24

Configuring NICs

OpenShift Container Platform deploys with two networks:

- **provisioning**: The **provisioning** network is a non-routable network used for provisioning the underlying operating system on each node that is a part of the OpenShift Container Platform cluster.
- **baremetal**: The **baremetal** network is a routable network. You can use any NIC order to interface with the **baremetal** network, provided it is not the NIC specified in the **provisioningNetworkInterface** configuration setting or the NIC associated to a node's **bootMACAddress** configuration setting for the **provisioning** network.

While the cluster nodes can contain more than two NICs, the installation process only focuses on the first two NICs. For example:

NIC	Network	VLAN
NIC1	provisioning	<provisioning_vlan>
NIC2	baremetal	<baremetal_vlan>

In the previous example, NIC1 on all control plane and worker nodes connects to the non-routable network (**provisioning**) that is only used for the installation of the OpenShift Container Platform cluster. NIC2 on all control plane and worker nodes connects to the routable **baremetal** network.

PXE	Boot order
NIC1 PXE-enabled provisioning network	1
NIC2 baremetal network.	2

**NOTE**

Ensure PXE is enabled on the NIC used for the **provisioning** network and is disabled on all other NICs.

Configuring canonical names

Clients access the OpenShift Container Platform cluster nodes over the **baremetal** network. Configure IBM Cloud subdomains or subzones where the canonical name extension is the cluster name.

```
<cluster_name>.<domain>
```

For example:

```
test-cluster.example.com
```

Creating DNS entries

You must create DNS **A** record entries resolving to unused IP addresses on the public subnet for the following:

Usage	Host Name	IP
API	api.<cluster_name>.<domain>	<ip>
Ingress LB (apps)	*.apps.<cluster_name>.<domain>	<ip>

Control plane and worker nodes already have DNS entries after provisioning.

The following table provides an example of fully qualified domain names. The API and Nameserver addresses begin with canonical name extensions. The host names of the control plane and worker nodes are examples, so you can use any host naming convention you prefer.

Usage	Host Name	IP
API	api.<cluster_name>.<domain>	<ip>
Ingress LB (apps)	*.apps.<cluster_name>.<domain>	<ip>
Provisioner node	provisioner.<cluster_name>.<domain>	<ip>
Master-0	openshift-master-0.<cluster_name>.<domain>	<ip>
Master-1	openshift-master-1.<cluster_name>.<domain>	<ip>
Master-2	openshift-master-2.<cluster_name>.<domain>	<ip>

Usage	Host Name	IP
Worker-0	openshift-worker-0. <cluster_name>.<domain>	<ip>
Worker-1	openshift-worker-1. <cluster_name>.<domain>	<ip>
Worker-n	openshift-worker-n. <cluster_name>.<domain>	<ip>

OpenShift Container Platform includes functionality that uses cluster membership information to generate **A** records. This resolves the node names to their IP addresses. After the nodes are registered with the API, the cluster can disperse node information without using CoreDNS-mDNS. This eliminates the network traffic associated with multicast DNS.



IMPORTANT

After provisioning the IBM Cloud nodes, you must create a DNS entry for the **api.<cluster_name>.<domain>** domain name on the external DNS because removing CoreDNS causes the local entry to disappear. Failure to create a DNS record for the **api.<cluster_name>.<domain>** domain name in the external DNS server prevents worker nodes from joining the cluster.

Network Time Protocol (NTP)

Each OpenShift Container Platform node in the cluster must have access to an NTP server. OpenShift Container Platform nodes use NTP to synchronize their clocks. For example, cluster nodes use SSL certificates that require validation, which might fail if the date and time between the nodes are not in sync.



IMPORTANT

Define a consistent clock date and time format in each cluster node's BIOS settings, or installation might fail.

Configure a DHCP server

IBM Cloud Bare Metal (Classic) does not run DHCP on the public or private VLANs. After provisioning IBM Cloud nodes, you must set up a DHCP server for the public VLAN, which corresponds to OpenShift Container Platform's **baremetal** network.



NOTE

The IP addresses allocated to each node do not need to match the IP addresses allocated by the IBM Cloud Bare Metal (Classic) provisioning system.

See the "Configuring the public subnet" section for details.

Ensure BMC access privileges

The "Remote management" page for each node on the dashboard contains the node's intelligent platform management interface (IPMI) credentials. The default IPMI privileges prevent the user from

making certain boot target changes. You must change the privilege level to **OPERATOR** so that Ironic can make those changes.

In the `install-config.yaml` file, add the `privilegelevel` parameter to the URLs used to configure each BMC. See the "Configuring the `install-config.yaml` file" section for additional details. For example:

```
ipmi://<IP>:<port>?privilegelevel=OPERATOR
```

Alternatively, contact IBM Cloud support and request that they increase the IPMI privileges to **ADMINISTRATOR** for each node.

Create bare metal servers

Create bare metal servers in the [IBM Cloud dashboard](#) by navigating to **Create resource** → **Bare Metal Servers for Classic**.

Alternatively, you can create bare metal servers with the `ibmcloud` CLI utility. For example:

```
$ ibmcloud sl hardware create --hostname <SERVERNAME> \  
    --domain <DOMAIN> \  
    --size <SIZE> \  
    --os <OS-TYPE> \  
    --datacenter <DC-NAME> \  
    --port-speed <SPEED> \  
    --billing <BILLING>
```

See [Installing the stand-alone IBM Cloud CLI](#) for details on installing the IBM Cloud CLI.



NOTE

IBM Cloud servers might take 3-5 hours to become available.

17.2. SETTING UP THE ENVIRONMENT FOR AN OPENSIFT CONTAINER PLATFORM INSTALLATION

17.2.1. Preparing the provisioner node on IBM Cloud Bare Metal (Classic) infrastructure

Perform the following steps to prepare the provisioner node.

Procedure

1. Log in to the provisioner node via `ssh`.
2. Create a non-root user (`kni`) and provide that user with `sudo` privileges:

```
# useradd kni
```

```
# passwd kni
```

```
# echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
```

```
# chmod 0440 /etc/sudoers.d/kni
```


-
- 3. Create an **ssh** key for the new user:

```
# su - kni -c "ssh-keygen -f /home/kni/.ssh/id_rsa -N ""
```

- 4. Log in as the new user on the provisioner node:

```
# su - kni
```

- 5. Use Red Hat Subscription Manager to register the provisioner node:

```
$ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
```

```
$ sudo subscription-manager repos --enable=rhel-8-for-x86_64-appstream-rpms \
--enable=rhel-8-for-x86_64-baseos-rpms
```



NOTE

For more information about Red Hat Subscription Manager, see [Using and Configuring Red Hat Subscription Manager](#).

- 6. Install the following packages:

```
$ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
```

- 7. Modify the user to add the **libvirt** group to the newly created user:

```
$ sudo usermod --append --groups libvirt kni
```

- 8. Start **firewalld**:

```
$ sudo systemctl start firewalld
```

- 9. Enable **firewalld**:

```
$ sudo systemctl enable firewalld
```

- 10. Start the **http** service:

```
$ sudo firewall-cmd --zone=public --add-service=http --permanent
```

```
$ sudo firewall-cmd --reload
```

- 11. Start and enable the **libvirtd** service:

```
$ sudo systemctl enable libvirtd --now
```

- 12. Set the ID of the provisioner node:

```
$ PRVN_HOST_ID=<ID>
```

You can view the ID with the following **ibmcloud** command:

```
$ ibmcloud sl hardware list
```

13. Set the ID of the public subnet:

```
$ PUBLICSUBNETID=<ID>
```

You can view the ID with the following **ibmcloud** command:

```
$ ibmcloud sl subnet list
```

14. Set the ID of the private subnet:

```
$ PRIVSUBNETID=<ID>
```

You can view the ID with the following **ibmcloud** command:

```
$ ibmcloud sl subnet list
```

15. Set the provisioner node public IP address:

```
$ PRVN_PUB_IP=$(ibmcloud sl hardware detail $PRVN_HOST_ID --output JSON | jq  
.primaryIpAddress -r)
```

16. Set the CIDR for the public network:

```
$ PUBLICCIDR=$(ibmcloud sl subnet detail $PUBLICSUBNETID --output JSON | jq .cidr)
```

17. Set the IP address and CIDR for the public network:

```
$ PUB_IP_CIDR=$PRVN_PUB_IP/$PUBLICCIDR
```

18. Set the gateway for the public network:

```
$ PUB_GATEWAY=$(ibmcloud sl subnet detail $PUBLICSUBNETID --output JSON | jq  
.gateway -r)
```

19. Set the private IP address of the provisioner node:

```
$ PRVN_PRIV_IP=$(ibmcloud sl hardware detail $PRVN_HOST_ID --output JSON | \  
jq .primaryBackendIpAddress -r)
```

20. Set the CIDR for the private network:

```
$ PRIVCIDR=$(ibmcloud sl subnet detail $PRIVSUBNETID --output JSON | jq .cidr)
```

21. Set the IP address and CIDR for the private network:

```
$ PRIV_IP_CIDR=$PRVN_PRIV_IP/$PRIVCIDR
```

22. Set the gateway for the private network:

```
$ PRIV_GATEWAY=$(ibmcloud sl subnet detail $PRIVSUBNETID --output JSON | jq
.gateway -r)
```

23. Set up the bridges for the **baremetal** and **provisioning** networks:

```
$ sudo nohup bash -c "
  nmcli --get-values UUID con show | xargs -n 1 nmcli con delete
  nmcli connection add ifname provisioning type bridge con-name provisioning
  nmcli con add type bridge-slave ifname eth1 master provisioning
  nmcli connection add ifname baremetal type bridge con-name baremetal
  nmcli con add type bridge-slave ifname eth2 master baremetal
  nmcli connection modify baremetal ipv4.addresses $PUB_IP_CIDR ipv4.method manual
  ipv4.gateway $PUB_GATEWAY
  nmcli connection modify provisioning ipv4.addresses 172.22.0.1/24,$PRIV_IP_CIDR
  ipv4.method manual
  nmcli connection modify provisioning +ipv4.routes \"10.0.0.0/8 $PRIV_GATEWAY\"
  nmcli con down baremetal
  nmcli con up baremetal
  nmcli con down provisioning
  nmcli con up provisioning
  init 6
"
```



NOTE

For **eth1** and **eth2**, substitute the appropriate interface name, as needed.

24. If required, SSH back into the **provisioner** node:

```
# ssh kni@provisioner.<cluster-name>.<domain>
```

25. Verify the connection bridges have been properly created:

```
$ sudo nmcli con show
```

Example output

```
NAME          UUID                                TYPE  DEVICE
baremetal     4d5133a5-8351-4bb9-bfd4-3af264801530 bridge baremetal
provisioning  43942805-017f-4d7d-a2c2-7cb3324482ed bridge provisioning
virbr0        d9bca40f-eee1-410b-8879-a2d4bb0465e7 bridge virbr0
bridge-slave-eth1 76a8ed50-c7e5-4999-b4f6-6d9014dd0812 ethernet eth1
bridge-slave-eth2 f31c3353-54b7-48de-893a-02d2b34c4736 ethernet eth2
```

26. Create a **pull-secret.txt** file:

```
$ vim pull-secret.txt
```

In a web browser, navigate to [Install on Bare Metal with user-provisioned infrastructure](#). In step 1, click **Download pull secret**. Paste the contents into the **pull-secret.txt** file and save the contents in the **kni** user's home directory.

17.2.2. Configuring the public subnet

All of the OpenShift Container Platform cluster nodes must be on the public subnet. IBM Cloud® Bare Metal (Classic) does not provide a DHCP server on the subnet. Set it up separately on the provisioner node.

You must reset the BASH variables defined when preparing the provisioner node. Rebooting the provisioner node after preparing it will delete the BASH variables previously set.

Procedure

1. Install **dnsmasq**:

```
$ sudo dnf install dnsmasq
```

2. Open the **dnsmasq** configuration file:

```
$ sudo vi /etc/dnsmasq.conf
```

3. Add the following configuration to the **dnsmasq** configuration file:

```
interface=baremetal
except-interface=lo
bind-dynamic
log-dhcp

dhcp-range=<ip_addr>,<ip_addr>,<pub_cidr> 1
dhcp-option=baremetal,121,0.0.0.0/0,<pub_gateway>,<prvn_priv_ip>,<prvn_pub_ip> 2
dhcp-hostsfile=/var/lib/dnsmasq/dnsmasq.hostsfile
```

- 1 Set the DHCP range. Replace both instances of **<ip_addr>** with one unused IP address from the public subnet so that the **dhcp-range** for the **baremetal** network begins and ends with the same the IP address. Replace **<pub_cidr>** with the CIDR of the public subnet.
- 2 Set the DHCP option. Replace **<pub_gateway>** with the IP address of the gateway for the **baremetal** network. Replace **<prvn_priv_ip>** with the IP address of the provisioner node's private IP address on the **provisioning** network. Replace **<prvn_pub_ip>** with the IP address of the provisioner node's public IP address on the **baremetal** network.

To retrieve the value for **<pub_cidr>**, execute:

```
$ ibmcloud sl subnet detail <publicsubnetid> --output JSON | jq .cidr
```

Replace **<publicsubnetid>** with the ID of the public subnet.

To retrieve the value for **<pub_gateway>**, execute:

```
$ ibmcloud sl subnet detail <publicsubnetid> --output JSON | jq .gateway -r
```

Replace **<publicsubnetid>** with the ID of the public subnet.

To retrieve the value for `<prvn_priv_ip>`, execute:

```
$ ibmcloud sl hardware detail <id> --output JSON | \
jq .primaryBackendIpAddress -r
```

Replace `<id>` with the ID of the provisioner node.

To retrieve the value for `<prvn_pub_ip>`, execute:

```
$ ibmcloud sl hardware detail <id> --output JSON | jq .primaryIpAddress -r
```

Replace `<id>` with the ID of the provisioner node.

- Obtain the list of hardware for the cluster:

```
$ ibmcloud sl hardware list
```

- Obtain the MAC addresses and IP addresses for each node:

```
$ ibmcloud sl hardware detail <id> --output JSON | \
jq '.networkComponents[] | \
"\(.primaryIpAddress) \(.macAddress)" | grep -v null
```

Replace `<id>` with the ID of the node.

Example output

```
"10.196.130.144 00:e0:ed:6a:ca:b4"
"141.125.65.215 00:e0:ed:6a:ca:b5"
```

Make a note of the MAC address and IP address of the public network. Make a separate note of the MAC address of the private network, which you will use later in the `install-config.yaml` file. Repeat this procedure for each node until you have all the public MAC and IP addresses for the public **baremetal** network, and the MAC addresses of the private **provisioning** network.

- Add the MAC and IP address pair of the public **baremetal** network for each node into the `dnsmasq.hostsfile` file:

```
$ sudo vim /var/lib/dnsmasq/dnsmasq.hostsfile
```

Example input

```
00:e0:ed:6a:ca:b5,141.125.65.215,master-0
<mac>,<ip>,master-1
<mac>,<ip>,master-2
<mac>,<ip>,worker-0
<mac>,<ip>,worker-1
...
```

Replace `<mac>`,`<ip>` with the public MAC address and public IP address of the corresponding node name.

- Start **dnsmasq**:

```
$ sudo systemctl start dnsmasq
```

8. Enable **dnsmasq** so that it starts when booting the node:

```
$ sudo systemctl enable dnsmasq
```

9. Verify **dnsmasq** is running:

```
$ sudo systemctl status dnsmasq
```

Example output

```
● dnsmasq.service - DNS caching server.  
Loaded: loaded (/usr/lib/systemd/system/dnsmasq.service; enabled; vendor preset: disabled)  
Active: active (running) since Tue 2021-10-05 05:04:14 CDT; 49s ago  
Main PID: 3101 (dnsmasq)  
Tasks: 1 (limit: 204038)  
Memory: 732.0K  
CGroup: /system.slice/dnsmasq.service  
└─3101 /usr/sbin/dnsmasq -k
```

10. Open ports **53** and **67** with UDP protocol:

```
$ sudo firewall-cmd --add-port 53/udp --permanent
```

```
$ sudo firewall-cmd --add-port 67/udp --permanent
```

11. Add **provisioning** to the external zone with masquerade:

```
$ sudo firewall-cmd --change-zone=provisioning --zone=external --permanent
```

This step ensures network address translation for IPMI calls to the management subnet.

12. Reload the **firewalld** configuration:

```
$ sudo firewall-cmd --reload
```

17.2.3. Retrieving the OpenShift Container Platform installer

Use the **stable-4.x** version of the installation program and your selected architecture to deploy the generally available stable version of OpenShift Container Platform:

```
$ export VERSION=stable-4.12
```

```
$ export RELEASE_ARCH=<architecture>
```

```
$ export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-  
v4/$RELEASE_ARCH/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print  
$3}')
```

17.2.4. Extracting the OpenShift Container Platform installer

After retrieving the installer, the next step is to extract it.

Procedure

1. Set the environment variables:

```
$ export cmd=openshift-baremetal-install
```

```
$ export pullsecret_file=~/.pull-secret.txt
```

```
$ export extract_dir=$(pwd)
```

2. Get the **oc** binary:

```
$ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

3. Extract the installer:

```
$ sudo cp oc /usr/local/bin
```

```
$ oc adm release extract --registry-config "${pullsecret_file}" --command=$cmd --to "${extract_dir}" ${RELEASE_IMAGE}
```

```
$ sudo cp openshift-baremetal-install /usr/local/bin
```

17.2.5. Configuring the install-config.yaml file

The **install-config.yaml** file requires some additional details. Most of the information is teaching the installer and the resulting cluster enough about the available IBM Cloud® Bare Metal (Classic) hardware so that it is able to fully manage it. The material difference between installing on bare metal and installing on IBM Cloud Bare Metal (Classic) is that you must explicitly set the privilege level for IPMI in the BMC section of the **install-config.yaml** file.

Procedure

1. Configure **install-config.yaml**. Change the appropriate variables to match the environment, including **pullSecret** and **sshKey**.

```
apiVersion: v1
baseDomain: <domain>
metadata:
  name: <cluster_name>
networking:
  machineNetwork:
    - cidr: <public-cidr>
    networkType: OVNKubernetes
compute:
  - name: worker
    replicas: 2
```

```

controlPlane:
  name: master
  replicas: 3
  platform:
    baremetal: {}
platform:
  baremetal:
    apiVIP: <api_ip>
    ingressVIP: <wildcard_ip>
    provisioningNetworkInterface: <NIC1>
    provisioningNetworkCIDR: <CIDR>
  hosts:
    - name: openshift-master-0
      role: master
      bmc:
        address: ipmi://10.196.130.145?privilegelevel=OPERATOR 1
        username: root
        password: <password>
        bootMACAddress: 00:e0:ed:6a:ca:b4 2
      rootDeviceHints:
        deviceName: "/dev/sda"
    - name: openshift-worker-0
      role: worker
      bmc:
        address: ipmi://<out-of-band-ip>?privilegelevel=OPERATOR 3
        username: <user>
        password: <password>
        bootMACAddress: <NIC1_mac_address> 4
      rootDeviceHints:
        deviceName: "/dev/sda"
  pullSecret: '<pull_secret>'
  sshKey: '<ssh_pub_key>'

```

1 3 The **bmc.address** provides a **privilegelevel** configuration setting with the value set to **OPERATOR**. This is required for IBM Cloud Bare Metal (Classic) infrastructure.

2 4 Add the MAC address of the private **provisioning** network NIC for the corresponding node.



NOTE

You can use the **ibmcloud** command-line utility to retrieve the password.

```

$ ibmcloud sl hardware detail <id> --output JSON | \
  jq ""(.networkManagementIpAddress)
  (.remoteManagementAccounts[0].password)""

```

Replace **<id>** with the ID of the node.

2. Create a directory to store the cluster configuration:

```
$ mkdir ~/clusterconfigs
```


- Copy the **install-config.yaml** file into the directory:

```
$ cp install-config.yaml ~/clusterconfig
```

- Ensure all bare metal nodes are powered off prior to installing the OpenShift Container Platform cluster:

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management_server_ip> power off
```

- Remove old bootstrap resources if any are left over from a previous deployment attempt:

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

17.2.6. Additional install-config parameters

See the following tables for the required parameters, the **hosts** parameter, and the **bmc** parameter for the **install-config.yaml** file.

Table 17.2. Required parameters

Parameters	Default	Description
baseDomain		The domain name for the cluster. For example, example.com .
bootMode	UEFI	The boot mode for a node. Options are legacy , UEFI , and UEFISecureBoot . If bootMode is not set, Ironic sets it while inspecting the node.
bootstrapExternalStaticIP		The static IP address for the bootstrap VM. You must set this value when deploying a cluster with static IP addresses when there is no DHCP server on the bare-metal network.
bootstrapExternalStaticGateway		The static IP address of the gateway for the bootstrap VM. You must set this value when deploying a cluster with static IP addresses when there is no DHCP server on the bare-metal network.
sshKey		The sshKey configuration setting contains the key in the <code>~/.ssh/id_rsa.pub</code> file required to access the control plane nodes and worker nodes. Typically, this key is from the provisioner node.

Parameters	Default	Description
pullSecret		The pullSecret configuration setting contains a copy of the pull secret downloaded from the Install OpenShift on Bare Metal page when preparing the provisioner node.
metadata: name:		The name to be given to the OpenShift Container Platform cluster. For example, openshift .
networking: machineNetwork: - cidr:		The public CIDR (Classless Inter-Domain Routing) of the external network. For example, 10.0.0.0/24 .
compute: - name: worker		The OpenShift Container Platform cluster requires a name be provided for worker (or compute) nodes even if there are zero nodes.
compute: replicas: 2		Replicas sets the number of worker (or compute) nodes in the OpenShift Container Platform cluster.
controlPlane: name: master		The OpenShift Container Platform cluster requires a name for control plane (master) nodes.
controlPlane: replicas: 3		Replicas sets the number of control plane (master) nodes included as part of the OpenShift Container Platform cluster.
provisioningNetwork Interface		The name of the network interface on nodes connected to the provisioning network. For OpenShift Container Platform 4.9 and later releases, use the bootMACAddress configuration setting to enable Ironic to identify the IP address of the NIC instead of using the provisioningNetworkInterface configuration setting to identify the name of the NIC.
defaultMachinePlatform		The default configuration used for machine pools without a platform configuration.

Parameters	Default	Description
apiVIPs		<p>(Optional) The virtual IP address for Kubernetes API communication.</p> <p>This setting must either be provided in the install-config.yaml file as a reserved IP from the MachineNetwork or preconfigured in the DNS so that the default name resolves correctly. Use the virtual IP address and not the FQDN when adding a value to the apiVIPs configuration setting in the install-config.yaml file. The primary IP address must be from the IPv4 network when using dual stack networking. If not set, the installation program uses api.<cluster_name>.<base_domain> to derive the IP address from the DNS.</p> <div data-bbox="660 689 767 1037" style="float: left; width: 60px; height: 155px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, #ccc 2px, #ccc 4px);"></div> <p style="margin-left: 60px;">NOTE</p> <p style="margin-left: 60px;">Before OpenShift Container Platform 4.12, the cluster installation program only accepted an IPv4 address or an IPv6 address for the apiVIP configuration setting. From OpenShift Container Platform 4.12 or later, the apiVIP configuration setting is deprecated. Instead, use a list format for the apiVIPs configuration setting to specify an IPv4 address, an IPv6 address or both IP address formats.</p>
disableCertificateVerification	False	<p>redfish and redfish-virtualmedia need this parameter to manage BMC addresses. The value should be True when using a self-signed certificate for BMC addresses.</p>
ingressVIPs		<p>(Optional) The virtual IP address for ingress traffic.</p> <p>This setting must either be provided in the install-config.yaml file as a reserved IP from the MachineNetwork or preconfigured in the DNS so that the default name resolves correctly. Use the virtual IP address and not the FQDN when adding a value to the ingressVIPs configuration setting in the install-config.yaml file. The primary IP address must be from the IPv4 network when using dual stack networking. If not set, the installation program uses test.apps.<cluster_name>.<base_domain> to derive the IP address from the DNS.</p> <div data-bbox="660 1715 767 2063" style="float: left; width: 60px; height: 155px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, #ccc 2px, #ccc 4px);"></div> <p style="margin-left: 60px;">NOTE</p> <p style="margin-left: 60px;">Before OpenShift Container Platform 4.12, the cluster installation program only accepted an IPv4 address or an IPv6 address for the ingressVIP configuration setting. In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a list format for the ingressVIPs configuration setting to specify an IPv4 addresses, an IPv6 addresses or both IP address formats.</p>

Table 17.3. Optional Parameters

Parameters	Default	Description
provisioningDHCPRange	172.22.0.10,172.22.0.100	Defines the IP range for nodes on the provisioning network.
provisioningNetworkCIDR	172.22.0.0/24	The CIDR for the network to use for provisioning. This option is required when not using the default address range on the provisioning network.
clusterProvisioningIP	The third IP address of the provisioningNetworkCIDR .	The IP address within the cluster where the provisioning services run. Defaults to the third IP address of the provisioning subnet. For example, 172.22.0.3 .
bootstrapProvisioningIP	The second IP address of the provisioningNetworkCIDR .	The IP address on the bootstrap VM where the provisioning services run while the installer is deploying the control plane (master) nodes. Defaults to the second IP address of the provisioning subnet. For example, 172.22.0.2 or 2620:52:0:1307::2 .
externalBridge	baremetal	The name of the bare-metal bridge of the hypervisor attached to the bare-metal network.
provisioningBridge	provisioning	The name of the provisioning bridge on the provisioner host attached to the provisioning network.
architecture		Defines the host architecture for your cluster. Valid values are amd64 or arm64 .
defaultMachinePlatform		The default configuration used for machine pools without a platform configuration.
bootstrapOSImage		A URL to override the default operating system image for the bootstrap node. The URL must contain a SHA-256 hash of the image. For example: <a href="https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256>">https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256> .


Parameters	Default	Description
provisioningNetwork		<p>The provisioningNetwork configuration setting determines whether the cluster uses the provisioning network. If it does, the configuration setting also determines if the cluster manages the network.</p> <p>Disabled: Set this parameter to Disabled to disable the requirement for a provisioning network. When set to Disabled, you must only use virtual media based provisioning, or bring up the cluster using the assisted installer. If Disabled and using power management, BMCs must be accessible from the bare-metal network. If Disabled, you must provide two IP addresses on the bare-metal network that are used for the provisioning services.</p> <p>Managed: Set this parameter to Managed, which is the default, to fully manage the provisioning network, including DHCP, TFTP, and so on.</p> <p>Unmanaged: Set this parameter to Unmanaged to enable the provisioning network but take care of manual configuration of DHCP. Virtual media provisioning is recommended but PXE is still available if required.</p>
httpProxy		Set this parameter to the appropriate HTTP proxy used within your environment.
httpsProxy		Set this parameter to the appropriate HTTPS proxy used within your environment.
noProxy		Set this parameter to the appropriate list of exclusions for proxy usage within your environment.

Hosts

The **hosts** parameter is a list of separate bare metal assets used to build the cluster.

Table 17.4. Hosts

Name	Default	Description
name		The name of the BareMetalHost resource to associate with the details. For example, openshift-master-0 .
role		The role of the bare metal node. Either master or worker .
bmc		Connection details for the baseboard management controller. See the BMC addressing section for additional details.

Name	Default	Description
bootMACAddress		<p>The MAC address of the NIC that the host uses for the provisioning network. Ironic retrieves the IP address using the bootMACAddress configuration setting. Then, it binds to the host.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>You must provide a valid MAC address from the host if you disabled the provisioning network.</p> </div> </div>
networkConfig		<p>Set this optional parameter to configure the network interface of a host. See "(Optional) Configuring host network interfaces" for additional details.</p>

17.2.7. Root device hints

The **rootDeviceHints** parameter enables the installer to provision the Red Hat Enterprise Linux CoreOS (RHCOS) image to a particular device. The installer examines the devices in the order it discovers them, and compares the discovered values with the hint values. The installer uses the first discovered device that matches the hint value. The configuration can combine multiple hints, but a device must match all hints for the installer to select it.

Table 17.5. Subfields

Subfield	Description
deviceName	A string containing a Linux device name like /dev/vda . The hint must match the actual value exactly.
hctl	A string containing a SCSI bus address like 0:0:0:0 . The hint must match the actual value exactly.
model	A string containing a vendor-specific device identifier. The hint can be a substring of the actual value.
vendor	A string containing the name of the vendor or manufacturer of the device. The hint can be a substring of the actual value.
serialNumber	A string containing the device serial number. The hint must match the actual value exactly.
minSizeGigabytes	An integer representing the minimum size of the device in gigabytes.

Subfield	Description
wwn	A string containing the unique storage identifier. The hint must match the actual value exactly.
wwnWithExtension	A string containing the unique storage identifier with the vendor extension appended. The hint must match the actual value exactly.
wwnVendorExtension	A string containing the unique vendor storage identifier. The hint must match the actual value exactly.
rotational	A boolean indicating whether the device should be a rotating disk (true) or not (false).

Example usage

```
- name: master-0
  role: master
  bmc:
    address: ipmi://10.10.0.3:6203
    username: admin
    password: redhat
  bootMACAddress: de:ad:be:ef:00:40
  rootDeviceHints:
    deviceName: "/dev/sda"
```

17.2.8. Creating the OpenShift Container Platform manifests

1. Create the OpenShift Container Platform manifests.

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
```

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
WARNING Discarding the OpenShift Manifest that was provided in the target directory
because its dependencies are dirty and it needs to be regenerated
```

17.2.9. Deploying the cluster via the OpenShift Container Platform installer

Run the OpenShift Container Platform installer:

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

17.2.10. Following the installation

During the deployment process, you can check the installation's overall status by issuing the **tail** command to the **.openshift_install.log** log file in the install directory folder:

```
$ tail -f /path/to/install-dir/.openshift_install.log
```


CHAPTER 18. INSTALLING WITH Z/VM ON IBM Z AND IBM LINUXONE

18.1. PREPARING TO INSTALL WITH Z/VM ON IBM Z AND IBM(R) LINUXONE

18.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

18.1.2. Choosing a method to install OpenShift Container Platform with z/VM on IBM Z or IBM(R) LinuxONE

You can install a cluster with z/VM on IBM Z or IBM® LinuxONE infrastructure that you provision, by using one of the following methods:

- **Installing a cluster with z/VM on IBM Z and IBM® LinuxONE** You can install OpenShift Container Platform with z/VM on IBM Z or IBM® LinuxONE infrastructure that you provision.
- **Installing a cluster with z/VM on IBM Z and IBM® LinuxONE in a restricted network** You can install OpenShift Container Platform with z/VM on IBM Z or IBM® LinuxONE infrastructure that you provision in a restricted or disconnected network, by using an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

18.2. INSTALLING A CLUSTER WITH Z/VM ON IBM Z AND IBM(R) LINUXONE

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Z or IBM® LinuxONE infrastructure that you provision.



NOTE

While this document refers only to IBM Z, all information in it also applies to IBM® LinuxONE.



IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

18.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- Before you begin the installation process, you must clean the installation directory. This ensures that the required installation files are created and updated during the installation process.
- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

18.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

18.2.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

18.2.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 18.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To improve high availability of your cluster, distribute the control plane machines over different z/VM instances on at least two physical machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

18.2.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 18.2. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. One physical core (IFL) provides two logical cores (threads) when SMT-2 is enabled. The hypervisor can provide two or more vCPUs.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

18.2.3.3. Minimum IBM Z system environment

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM z16 (all models), IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- IBM® LinuxONE Emperor 4, IBM® LinuxONE III (all models), IBM® LinuxONE Emperor II, IBM® LinuxONE Rockhopper II, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper



NOTE

Support for RHCOS functionality for IBM z13 all models, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

Hardware requirements

- The equivalent of six Integrated Facilities for Linux (IFL), which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.



NOTE

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.



IMPORTANT

Since the overall performance of the cluster can be impacted, the LPARs that are used to set up the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

Operating system requirements

- One instance of z/VM 7.2 or later

On your z/VM instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter

- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

Disk storage for the z/VM guest virtual machines

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV to ensure optimal performance.
- FCP attached disk storage

Storage / Main Memory

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

18.2.3.4. Preferred IBM Z system environment

Hardware requirements

- Three LPARS that each have the equivalent of six IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.
- HiperSockets, which are attached to a node either directly as a device or by bridging with one z/VM VSWITCH to be transparent to the z/VM guest. To directly connect HiperSockets to a node, you must set up a gateway to the external network via a RHEL 8 guest to bridge to the HiperSockets network.

Operating system requirements

- Two or three instances of z/VM 7.2 or later for high availability

On your z/VM instances, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines, one per z/VM instance.
- At least six guest virtual machines for OpenShift Container Platform compute machines, distributed across the z/VM instances.
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
- To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using the CP command **SET SHARE**. Do the same for infrastructure nodes, if they exist. See [SET SHARE](#) in IBM Documentation.

IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter

- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

Disk storage for the z/VM guest virtual machines

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV and High Performance FICON (zHPF) to ensure optimal performance.
- FCP attached disk storage

Storage / Main Memory

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

18.2.3.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- See [Bridging a HyperSockets LAN with a z/VM Virtual Switch](#) in IBM Documentation.
- See [Scaling HyperPAV alias devices on Linux guests on z/VM](#) for performance optimization.
- See [Topics in LPAR performance](#) for LPAR weight management and entitlements.
- [Recommended host practices for IBM Z & IBM® LinuxONE environments](#)

18.2.3.6. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an HTTP or HTTPS server to establish a network connection to download their Ignition config files.

The machines are configured with static IP addresses. No DHCP server is required. Ensure that the machines have persistent IP addresses and hostnames.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

18.2.3.6.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 18.3. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 18.4. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 18.5. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

Additional resources

- [Configuring chrony time service](#)

18.2.3.7. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines


Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 18.6. Required DNS records

Component	Record	Description
-----------	--------	-------------

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

18.2.3.7.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 18.1. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.

- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 18.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

18.2.3.8. Load balancing requirements for user-provisioned infrastructure

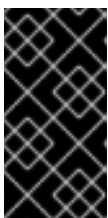
Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 18.7. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 18.8. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

18.2.3.8.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 18.3. Sample API and application Ingress load balancer configuration

```
global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request  10s
  timeout queue      1m
  timeout connect    10s
  timeout client     1m
```

```

timeout server      1m
timeout http-keep-alive 10s
timeout check      10s
maxconn            3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

18.2.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, preparing a web server for the Ignition files, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. Set up static IP addresses.
2. Set up an HTTP or HTTPS server to provide Ignition files to the cluster nodes.
3. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
4. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

5. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
6. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
7. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

18.2.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1 Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

-
- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.

- a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

18.2.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

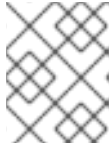
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

18.2.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on your provisioning machine.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

18.2.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:
 -

```
$ tar xvf <file>
```

- Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the appropriate version from the **Version** drop-down list.
- Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
- Unzip the archive with a ZIP program.
- Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the appropriate version from the **Version** drop-down list.
- Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

18.2.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

18.2.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

18.2.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 18.9. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

18.2.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

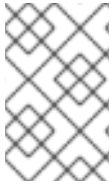
- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.



```
networking:
  clusterNetwork:
```

- cidr: 10.128.0.0/14
hostPrefix: 23
- cidr: fd00:10:128::/56
hostPrefix: 64
- serviceNetwork:
- 172.30.0.0/16
- fd00:172:16::/112

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 18.10. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.network Type	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example:  networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .

Parameter	Description	Values
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


18.2.9.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 18.11. Optional parameters

Parameter	Description	Values
-----------	-------------	--------



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String

Parameter	Description	Values
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.

Parameter	Description	Values
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1635" style="background-color: #f0f0f0; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

18.2.9.2. Sample install-config.yaml file for IBM Z

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4

```

```

architecture: s390x
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
  architecture: s390x
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not available on your OpenShift Container Platform nodes, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether on your OpenShift Container Platform nodes or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.

**NOTE**

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

**NOTE**

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Z infrastructure.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

18.2.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
```

```

proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

18.2.9.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a minimal three node cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

**NOTE**

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

**NOTE**

The preferred resource for control plane nodes is six vCPUs and 21 GB. For three control plane nodes this is the memory + vCPU equivalent of a minimum five-node cluster. You should back the three nodes, each installed on a 120 GB disk, with three IFLs that are SMT2 enabled. The minimum tested setup is three vCPUs and 10 GB on a 120 GB disk for each control plane node.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.

- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

18.2.10. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

18.2.10.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 18.12. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 18.13. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 18.14. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 18.15. `ovnKubernetesConfig` object

Field	Type	Description
<code>mtu</code>	<code>integer</code>	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
<code>genevePort</code>	<code>integer</code>	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
<code>ipsecConfig</code>	<code>object</code>	Specify an empty object to enable IPsec encryption.
<code>policyAuditConfig</code>	<code>object</code>	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
<code>gatewayConfig</code>	<code>object</code>	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 18.16. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 18.17. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

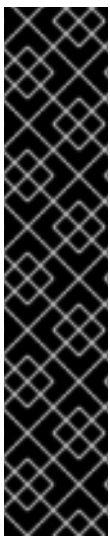
Table 18.18. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

18.2.11. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**NOTE**

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program runs on s390x only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

**IMPORTANT**

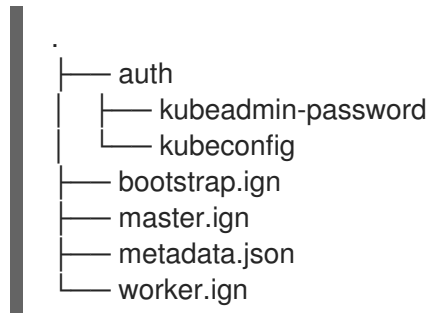
When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```


- 1 For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:



18.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Z infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on z/VM guest virtual machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS z/VM guest virtual machines have rebooted.

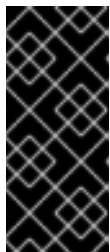
Complete the following steps to create the machines.

Prerequisites

- An HTTP or HTTPS server running on your provisioning machine that is accessible to the machines you create.

Procedure

1. Log in to Linux on your provisioning machine.
2. Obtain the Red Hat Enterprise Linux CoreOS (RHCOS) kernel, initramfs, and rootfs files from the [RHCOS image mirror](#).



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate kernel, initramfs, and rootfs artifacts described in the following procedure.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- kernel: **rhcos-`<version>`-live-kernel-`<architecture>`**
- initramfs: **rhcos-`<version>`-live-initramfs.`<architecture>`.img**

- rootfs: **rhcos-`<version>`-live-rootfs.`<architecture>`.img**

**NOTE**

The rootfs image is the same for FCP and DASD.

3. Create parameter files. The following parameters are specific for a particular virtual machine:

- For **ip=**, specify the following seven entries:
 - i. The IP address for the machine.
 - ii. An empty string.
 - iii. The gateway.
 - iv. The netmask.
 - v. The machine host and domain name in the form **hostname.domainname**. Omit this value to let RHCOS decide.
 - vi. The network interface name. Omit this value to let RHCOS decide.
 - vii. If you use static IP addresses, specify **none**.
- For **coreos.inst.ignition_url=**, specify the Ignition file for the machine role. Use **bootstrap.ign**, **master.ign**, or **worker.ign**. Only HTTP and HTTPS protocols are supported.
- For **coreos.live.rootfs_url=**, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.
- For installations on DASD-type disks, complete the following tasks:
 - i. For **coreos.inst.install_dev=**, specify **dasda**.
 - ii. Use **rd.dasd=** to specify the DASD where RHCOS is to be installed.
 - iii. Leave all other parameters unchanged.
Example parameter file, **bootstrap-0.parm**, for the bootstrap machine:

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcplib.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

Write all options in the parameter file as a single line and make sure you have no newline characters.

- For installations on FCP-type disks, complete the following tasks:
 - i. Use **rd.zfcplib=<adapter>,<wwpn>,<lun>** to specify the FCP disk where RHCOS is to be

installed. For multipathing repeat this step for each additional path.



NOTE

When you install with multiple paths, you must enable multipathing directly after the installation, not at a later point in time, as this can cause problems.

- ii. Set the install device as: **coreos.inst.install_dev=sda**.



NOTE

If additional LUNs are configured with NPIV, FCP requires **zfcplib.allow_lun_scan=0**. If you must enable **zfcplib.allow_lun_scan=1** because you use a CSI driver, for example, you must configure your NPIV so that each node cannot access the boot partition of another node.

- iii. Leave all other parameters unchanged.



IMPORTANT

Additional postinstallation steps are required to fully enable multipathing. For more information, see "Enabling multipathing with kernel arguments on RHCOS" in *Post-installation machine configuration tasks*.

The following is an example parameter file **worker-1.parm** for a worker node with multipathing:

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=sda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcplib.allow_lun_scan=0 \
rd.zfcplib=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcplib=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcplib=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcplib=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

Write all options in the parameter file as a single line and make sure you have no newline characters.

4. Transfer the initramfs, kernel, parameter files, and RHCOS images to z/VM, for example with FTP. For details about how to transfer the files with FTP and boot from the virtual reader, see [Installing under Z/VM](#).
5. Punch the files to the virtual reader of the z/VM guest virtual machine that is to become your bootstrap node.
See [PUNCH](#) in IBM Documentation.

TIP

You can use the CP PUNCH command or, if you use Linux, the **vmur** command to transfer files between two z/VM guest virtual machines.

6. Log in to CMS on the bootstrap machine.
7. IPL the bootstrap machine from the reader:

```
$ ipl c
```

See [IPL](#) in IBM Documentation.

8. Repeat this procedure for the other machines in the cluster.

18.2.12.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

18.2.12.1.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.

**IMPORTANT**

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.

**NOTE**

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**

- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option. Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.

- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Always set option **fail_over_mac=1** in active-backup mode, to avoid problems when shared OSA/RoCE cards are used.

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

18.2.13. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided

through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
┌ $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1  
└ --log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
┌ INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...  
└ INFO API v1.25.0 up  
└ INFO Waiting up to 30m0s for bootstrapping to complete...  
└ INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

18.2.14. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

18.2.15. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 63m  v1.25.0
master-1  Ready   master 63m  v1.25.0
master-2  Ready   master 64m  v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

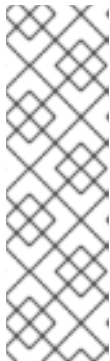
- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-mddf5	20m	system:node:master-01.example.com	Approved,Issued
csr-z5rln	16m	system:node:worker-21.example.com	Approved,Issued

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

18.2.16. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m

operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

18.2.16.1. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

18.2.16.1.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Z.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

- Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

- Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

- Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

- Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

18.2.16.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

18.2.17. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

- Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m

csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8  1/1
Running   0    5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

18.2.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service
- [How to generate SOSREPORT within OpenShift4 nodes without SSH](#) .

18.2.19. Next steps

- [Enabling multipathing with kernel arguments on RHCOS](#) .
- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

18.3. INSTALLING A CLUSTER WITH Z/VM ON IBM Z AND IBM(R) LINUXONE IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Z or IBM® LinuxONE infrastructure that you provision in a restricted network.



NOTE

While this document refers to only IBM Z, all information in it also applies to IBM® LinuxONE.



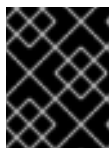
IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

18.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a mirror registry for installation in a restricted network](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.

- Before you begin the installation process, you must move or remove any existing installation files. This ensures that the required installation files are created and updated during the installation process.



IMPORTANT

Ensure that installation steps are done from a machine with access to the installation media.

- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

18.3.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

18.3.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

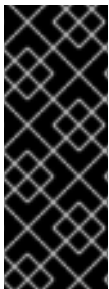
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

18.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

18.3.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

18.3.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 18.19. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.

**IMPORTANT**

To improve high availability of your cluster, distribute the control plane machines over different z/VM instances on at least two physical machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

18.3.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 18.20. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. One physical core (IFL) provides two logical cores (threads) when SMT-2 is enabled. The hypervisor can provide two or more vCPUs.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

18.3.4.3. Minimum IBM Z system environment

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM z16 (all models), IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- IBM® LinuxONE Emperor 4, IBM® LinuxONE III (all models), IBM® LinuxONE Emperor II, IBM® LinuxONE Rockhopper II, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper

**NOTE**

Support for RHCOS functionality for IBM z13 all models, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

Hardware requirements

- The equivalent of six Integrated Facilities for Linux (IFL), which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.



NOTE

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.



IMPORTANT

Since the overall performance of the cluster can be impacted, the LPARs that are used to set up the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

Operating system requirements

- One instance of z/VM 7.2 or later

On your z/VM instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

Disk storage for the z/VM guest virtual machines

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV to ensure optimal performance.
- FCP attached disk storage

Storage / Main Memory

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines

- 16 GB for the temporary OpenShift Container Platform bootstrap machine

18.3.4.4. Preferred IBM Z system environment

Hardware requirements

- Three LPARS that each have the equivalent of six IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.
- HiperSockets, which are attached to a node either directly as a device or by bridging with one z/VM VSWITCH to be transparent to the z/VM guest. To directly connect HiperSockets to a node, you must set up a gateway to the external network via a RHEL 8 guest to bridge to the HiperSockets network.

Operating system requirements

- Two or three instances of z/VM 7.2 or later for high availability

On your z/VM instances, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines, one per z/VM instance.
- At least six guest virtual machines for OpenShift Container Platform compute machines, distributed across the z/VM instances.
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
- To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using the CP command **SET SHARE**. Do the same for infrastructure nodes, if they exist. See [SET SHARE](#) in IBM Documentation.

IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

Disk storage for the z/VM guest virtual machines

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV and High Performance FICON (zHPF) to ensure optimal performance.
- FCP attached disk storage

Storage / Main Memory

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines

- 16 GB for the temporary OpenShift Container Platform bootstrap machine

18.3.4.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- See [Bridging a HyperSockets LAN with a z/VM Virtual Switch](#) in IBM Documentation.
- See [Scaling HyperPAV alias devices on Linux guests on z/VM](#) for performance optimization.
- See [Topics in LPAR performance](#) for LPAR weight management and entitlements.
- [Recommended host practices for IBM Z & IBM® LinuxONE environments](#)

18.3.4.6. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

18.3.4.6.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through

NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

18.3.4.6.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

Table 18.21. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 18.22. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 18.23. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

Additional resources

- [Configuring chrony time service](#)

18.3.4.7. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:


- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 18.24. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

18.3.4.7.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 18.4. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
```

```
compute1.ocp4.example.com. IN A 192.168.1.7 9
```

```
;  
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 18.5. Sample DNS zone database for reverse records

```
$TTL 1W  
@ IN SOA ns1.example.com. root (  
 2019070700 ; serial  
 3H ; refresh (3 hours)  
 30M ; retry (30 minutes)  
 2W ; expiry (2 weeks)  
 1W ) ; minimum (1 week)  
IN NS ns1.example.com.  
;  
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2  
;  
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3  
;  
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4  
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5  
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
```

```

;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF

```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

18.3.4.8. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

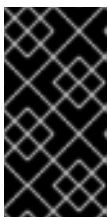


NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 18.25. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 18.26. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic

Port	Back-end machines (pool members)	Internal	External	Description
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

18.3.4.8.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running `setsebool -P haproxy_connect_any=1`.

Example 18.6. Sample API and application Ingress load balancer configuration

```
global
  log      127.0.0.1 local2
  pidfile /var/run/haproxy.pid
  maxconn 4000
  daemon
defaults
  mode            http
  log             global
  option          dontlognull
  option http-server-close
  option          redispatch
  retries         3
  timeout http-request 10s
  timeout queue   1m
  timeout connect 10s
  timeout client  1m
  timeout server  1m
  timeout http-keep-alive 10s
  timeout check   10s
  maxconn         3000
listen api-server-6443 1
```



```

bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1 Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2 4 The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3 Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltpu** on the HAProxy node.

18.3.5. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, preparing a web server for the Ignition files, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. Set up static IP addresses.
2. Set up an HTTP or HTTPS server to provide Ignition files to the cluster nodes.
3. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
4. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

5. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.

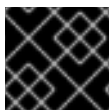
- b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
6. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
 7. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

18.3.6. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1 Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.

2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1 Provides the record name for the Kubernetes internal API.
- 2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

18.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

18.3.8. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

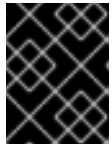
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

18.3.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

18.3.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 18.27. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .

Parameter	Description	Values
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

18.3.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.


```
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    - cidr: fd00:10:128::/56
      hostPrefix: 64
  serviceNetwork:
    - 172.30.0.0/16
    - fd00:172:16::/112
```




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 18.28. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .

Parameter	Description	Values
networking.clusterNetwork	<p>The IP address blocks for pods.</p> <p>The default value is 10.128.0.0/14 with a host prefix of /23.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	<p>Required if you use networking.clusterNetwork. An IP address block.</p> <p>An IPv4 network.</p>	<p>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</p>
networking.clusterNetwork.hostPrefix	<p>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.</p>	<p>A subnet prefix.</p> <p>The default value is 23.</p>
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p> <p>If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


18.3.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 18.29. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...

18.3.8.2. Sample install-config.yaml file for IBM Z

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
  architecture: s390x
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7

```


**NOTE**

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

**NOTE**

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Z infrastructure.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 17 Add the **additionalTrustBundle** parameter and value. The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.
- 18 Provide the **imageContentSources** section from the output of the command to mirror the repository.

18.3.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

18.3.8.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a minimal three node cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



NOTE

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.



NOTE

The preferred resource for control plane nodes is six vCPUs and 21 GB. For three control plane nodes this is the memory + vCPU equivalent of a minimum five-node cluster. You should back the three nodes, each installed on a 120 GB disk, with three IFLs that are SMT2 enabled. The minimum tested setup is three vCPUs and 10 GB on a 120 GB disk for each control plane node.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your

application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.

- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

18.3.9. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

18.3.9.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 18.30. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .


Field	Type	Description
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 18.31. **defaultNetwork** object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 18.32. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 18.33. ovnKubernetesConfig object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. This field cannot be changed after installation.	The default value is fd98::/48 .

Table 18.34. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 18.35. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 18.36. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

18.3.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**NOTE**

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program runs on s390x only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

**IMPORTANT**

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:


```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```

.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign

```

18.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Z infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on z/VM guest virtual machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS z/VM guest virtual machines have rebooted.

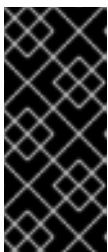
Complete the following steps to create the machines.

Prerequisites

- An HTTP or HTTPS server running on your provisioning machine that is accessible to the machines you create.

Procedure

1. Log in to Linux on your provisioning machine.
2. Obtain the Red Hat Enterprise Linux CoreOS (RHCOS) kernel, initramfs, and rootfs files from the [RHCOS image mirror](#).



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate kernel, initramfs, and rootfs artifacts described in the following procedure.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- kernel: **rhcos-<version>-live-kernel-<architecture>**

- initramfs: **rhcos-`<version>`-live-initramfs.`<architecture>`.img**
- rootfs: **rhcos-`<version>`-live-rootfs.`<architecture>`.img**

**NOTE**

The rootfs image is the same for FCP and DASD.

3. Create parameter files. The following parameters are specific for a particular virtual machine:

- For **ip=**, specify the following seven entries:
 - The IP address for the machine.
 - An empty string.
 - The gateway.
 - The netmask.
 - The machine host and domain name in the form **hostname.domainname**. Omit this value to let RHCOS decide.
 - The network interface name. Omit this value to let RHCOS decide.
 - If you use static IP addresses, specify **none**.
- For **coreos.inst.ignition_url=**, specify the Ignition file for the machine role. Use **bootstrap.ign**, **master.ign**, or **worker.ign**. Only HTTP and HTTPS protocols are supported.
- For **coreos.live.rootfs_url=**, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.
- For installations on DASD-type disks, complete the following tasks:
 - For **coreos.inst.install_dev=**, specify **dasda**.
 - Use **rd.dasd=** to specify the DASD where RHCOS is to be installed.
 - Leave all other parameters unchanged.
Example parameter file, **bootstrap-0.parm**, for the bootstrap machine:

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcpl.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

Write all options in the parameter file as a single line and make sure you have no newline characters.

- For installations on FCP-type disks, complete the following tasks:
 - i. Use **rd.zfcp=<adapter>,<wwpn>,<lun>** to specify the FCP disk where RHCOS is to be installed. For multipathing repeat this step for each additional path.

**NOTE**

When you install with multiple paths, you must enable multipathing directly after the installation, not at a later point in time, as this can cause problems.

- ii. Set the install device as: **coreos.inst.install_dev=sda**.

**NOTE**

If additional LUNs are configured with NPIV, FCP requires **zfcp.allow_lun_scan=0**. If you must enable **zfcp.allow_lun_scan=1** because you use a CSI driver, for example, you must configure your NPIV so that each node cannot access the boot partition of another node.

- iii. Leave all other parameters unchanged.

**IMPORTANT**

Additional postinstallation steps are required to fully enable multipathing. For more information, see "Enabling multipathing with kernel arguments on RHCOS" in *Post-installation machine configuration tasks*.

The following is an example parameter file **worker-1.parm** for a worker node with multipathing:

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=sda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.zfcp=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

Write all options in the parameter file as a single line and make sure you have no newline characters.

4. Transfer the initramfs, kernel, parameter files, and RHCOS images to z/VM, for example with FTP. For details about how to transfer the files with FTP and boot from the virtual reader, see [Installing under Z/VM](#).

- Punch the files to the virtual reader of the z/VM guest virtual machine that is to become your bootstrap node.
See [PUNCH](#) in IBM Documentation.

TIP

You can use the CP PUNCH command or, if you use Linux, the **vmur** command to transfer files between two z/VM guest virtual machines.

- Log in to CMS on the bootstrap machine.
- IPL the bootstrap machine from the reader:

```
$ ipl c
```

See [IPL](#) in IBM Documentation.

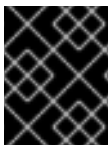
- Repeat this procedure for the other machines in the cluster.

18.3.11.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

18.3.11.1.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option. Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Always set option **fail_over_mac=1** in active-backup mode, to avoid problems when shared OSA/RoCE cards are used.

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]** *name* is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#) .

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

18.3.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

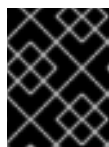
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

18.3.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container

Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

18.3.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master  63m  v1.25.0
master-1  Ready   master  63m  v1.25.0
master-2  Ready   master  64m  v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}
{{end}}{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   73m   v1.25.0
master-1  Ready   master   73m   v1.25.0
```

```

master-2 Ready   master 74m v1.25.0
worker-0 Ready   worker 11m v1.25.0
worker-1 Ready   worker 11m v1.25.0

```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

18.3.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m

node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

18.3.15.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

18.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

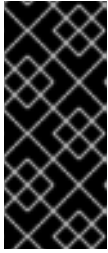
18.3.15.2.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Z.

- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

- To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

- Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

- Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

- Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

18.3.15.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

18.3.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```
NAMESPACE          NAME                                READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running    1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running    0    5m
...
```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.
4. Register your cluster on the [Cluster registration](#) page.

Additional resources

- [How to generate SOSREPORT within OpenShift Container Platform version 4 nodes without SSH.](#)

18.3.17. Next steps

- [Customize your cluster.](#)
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)

CHAPTER 19. INSTALLING WITH RHEL KVM ON IBM Z AND IBM LINUXONE

19.1. PREPARING TO INSTALL WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE

19.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

19.1.2. Choosing a method to install OpenShift Container Platform with RHEL KVM on IBM Z or IBM(R) LinuxONE

You can install a cluster with RHEL KVM on IBM Z or IBM® LinuxONE infrastructure that you provision, by using one of the following methods:

- **Installing a cluster with RHEL KVM on IBM Z and IBM® LinuxONE** You can install OpenShift Container Platform with KVM on IBM Z or IBM® LinuxONE infrastructure that you provision.
- **Installing a cluster with RHEL KVM on IBM Z and IBM® LinuxONE in a restricted network** You can install OpenShift Container Platform with RHEL KVM on IBM Z or IBM® LinuxONE infrastructure that you provision in a restricted or disconnected network, by using an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

19.2. INSTALLING A CLUSTER WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Z or IBM® LinuxONE infrastructure that you provision.



NOTE

While this document refers only to IBM Z, all information in it also applies to IBM® LinuxONE.



IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

19.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- Before you begin the installation process, you must clean the installation directory. This ensures that the required installation files are created and updated during the installation process.
- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

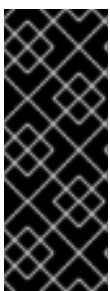
- You provisioned a RHEL Kernel Virtual Machine (KVM) system that is hosted on the logical partition (LPAR) and based on RHEL 8.4 or later. See [Red Hat Enterprise Linux 8 and 9 Life Cycle](#).

19.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

19.2.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

One or more KVM host machines based on RHEL 8.4 or later. Each RHEL KVM host machine must have libvirt installed and running. The virtual machines are provisioned under each RHEL KVM host machine.

19.2.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

Table 19.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To improve high availability of your cluster, distribute the control plane machines over different RHEL instances on at least two physical machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

See [Red Hat Enterprise Linux technology capabilities and limits](#) .

19.2.3.2. Network connectivity requirements

The OpenShift Container Platform installer creates the Ignition files, which are necessary for all the Red Hat Enterprise Linux CoreOS (RHCOS) virtual machines. The automated installation of OpenShift Container Platform is performed by the bootstrap machine. It starts the installation of OpenShift Container Platform on each node, starts the Kubernetes cluster, and then finishes. During this bootstrap, the virtual machine must have an established network connection either through a Dynamic Host Configuration Protocol (DHCP) server or static IP address.

19.2.3.3. IBM Z network connectivity requirements

To install on IBM Z under RHEL KVM, you need:

- A RHEL KVM host configured with an OSA or RoCE network adapter.
- Either a RHEL KVM host that is configured to use bridged networking in libvirt or MacVTap to connect the network to the guests.
See [Types of virtual network connections](#) .

19.2.3.4. Host machine resource requirements

The RHEL KVM host in your environment must meet the following requirements to host the virtual machines that you plan for the OpenShift Container Platform environment. See [Getting started with virtualization](#).

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM z16 (all models), IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- IBM® LinuxONE Emperor 4, IBM® LinuxONE III (all models), IBM® LinuxONE Emperor II, IBM® LinuxONE Rockhopper II, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper



NOTE

Support for RHCOS functionality for IBM z13 all models, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

19.2.3.5. Minimum IBM Z system environment

Hardware requirements

- The equivalent of six Integrated Facilities for Linux (IFL), which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.



NOTE

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.



IMPORTANT

Since the overall performance of the cluster can be impacted, the LPARs that are used to set up the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

Operating system requirements

- One LPAR running on RHEL 8.4 or later with KVM, which is managed by libvirt

On your RHEL KVM host, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

19.2.3.6. Minimum resource requirements

Each cluster virtual machine must meet the following minimum requirements:

Virtual Machine	Operating System	vCPU [1]	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. One physical core (IFL) provides two logical cores (threads) when SMT-2 is enabled. The hypervisor can provide two or more vCPUs.

19.2.3.7. Preferred IBM Z system environment

Hardware requirements

- Three LPARS that each have the equivalent of six IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.

Operating system requirements

- For high availability, two or three LPARs running on RHEL 8.4 or later with KVM, which are managed by libvirt.

On your RHEL KVM host, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines, distributed across the RHEL KVM host machines.
- At least six guest virtual machines for OpenShift Container Platform compute machines, distributed across the RHEL KVM host machines.
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
- To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using **cpu_shares**. Do the same for infrastructure nodes, if they exist. See [schedinfo](#) in IBM Documentation.

19.2.3.8. Preferred resource requirements

The preferred requirements for each cluster virtual machine are:

Virtual Machine	Operating System	vCPU	Virtual RAM	Storage
Bootstrap	RHCOS	4	16 GB	120 GB
Control plane	RHCOS	8	16 GB	120 GB
Compute	RHCOS	6	8 GB	120 GB

Virtual Machine

Operating System

vCPU

Virtual RAM

Storage

19.2.3.9. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- [Recommended host practices for IBM Z & IBM® LinuxONE environments](#)

19.2.3.10. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

19.2.3.10.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a

reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

19.2.3.10.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.



NOTE

The RHEL KVM host must be configured to use bridged networking in libvirt or MacVTap to connect the network to the virtual machines. The virtual machines must have access to the network, which is attached to the RHEL KVM host. Virtual Networks, for example network address translation (NAT), within KVM are not a supported configuration.

Table 19.2. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets

Protocol	Port	Description
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 19.3. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 19.4. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

19.2.3.11. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:


- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 19.5. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>
Routes	*.apps.<cluster_name>.<base_domain>..	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain>.. is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

Component	Record	Description
Control plane machines	<control_plane><n>. <cluster_name>. <base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>. <cluster_name>. <base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

19.2.3.11.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 19.1. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
```

```

;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 19.2. Sample DNS zone database for reverse records

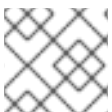
```
$TTL 1W
```

```

@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF

```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

19.2.3.12. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 19.6. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 19.7. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

19.2.3.12.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 19.3. Sample API and application Ingress load balancer configuration

```
global
log      127.0.0.1 local2
pidfile /var/run/haproxy.pid
maxconn 4000
daemon
defaults
```



```

mode          http
log           global
option       dontlognull
option http-server-close
option       redispatch
retries      3
timeout http-request 10s
timeout queue 1m
timeout connect 10s
timeout client 1m
timeout server 1m
timeout http-keep-alive 10s
timeout check 10s
maxconn      3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.

- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltpu** on the HAProxy node.

19.2.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

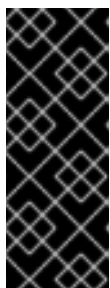
If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.

**NOTE**

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Choose to perform either a fast track installation of Red Hat Enterprise Linux CoreOS (RHCOS) or a full installation of Red Hat Enterprise Linux CoreOS (RHCOS). For the full installation, you must set up an HTTP or HTTPS server to provide Ignition files and install images to the cluster nodes. For the fast track installation an HTTP or HTTPS server is not required, however, a DHCP server is required. See sections "Fast-track installation: Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines" and "Full installation: Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines".
3. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
4. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

5. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
6. Validate your DNS configuration.

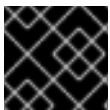
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
7. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

19.2.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
- a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal

API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1 Provides the record name for the Kubernetes internal API.
- 2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

19.2.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

■

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

19.2.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on your provisioning machine.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```


- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

19.2.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the architecture from the **Product Variant** drop-down list.
- Select the appropriate version from the **Version** drop-down list.
- Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
- Unpack the archive:

```
$ tar xvf <file>
```

- Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

- Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
- Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

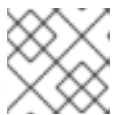
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

19.2.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

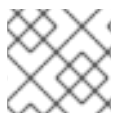


IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

19.2.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

19.2.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 19.8. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

19.2.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.


```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 19.9. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p> <p>If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


19.2.9.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 19.10. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1435 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

19.2.9.2. Sample install-config.yaml file for IBM Z

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
  architecture: s390x
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7

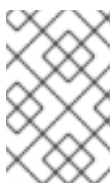
```

```

architecture: s390x
metadata:
  name: test 8
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14 9
      hostPrefix: 23 10
    networkType: OVNKubernetes 11
  serviceNetwork: 12
    - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1** The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2** **5** The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3** **6** Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not available on your OpenShift Container Platform nodes, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether on your OpenShift Container Platform nodes or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4** You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7** The number of control plane machines that you add to the cluster. Because the cluster uses these

- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Z infrastructure.



IMPORTANT

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

19.2.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

19.2.9.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a minimal three node cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



NOTE

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.



NOTE

The preferred resource for control plane nodes is six vCPUs and 21 GB. For three control plane nodes this is the memory + vCPU equivalent of a minimum five-node cluster. You should back the three nodes, each installed on a 120 GB disk, with three IFLs that are SMT2 enabled. The minimum tested setup is three vCPUs and 10 GB on a 120 GB disk for each control plane node.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

19.2.10. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

19.2.10.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 19.11. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.

Field	Type	Description
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 19.12. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 19.13. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 19.14. `ovnKubernetesConfig` object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 19.15. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 19.16. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 19.17. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

19.2.11. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**NOTE**

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program runs on s390x only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

**IMPORTANT**

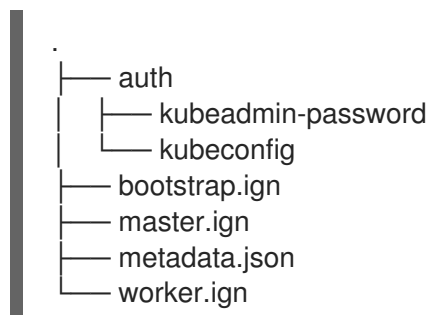
When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:



19.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Z infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) as Red Hat Enterprise Linux (RHEL) guest virtual machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

You can perform a fast-track installation of RHCOS that uses a prepackaged QEMU copy-on-write (QCOW2) disk image. Alternatively, you can perform a full installation on a new QCOW2 disk image.

To add further security to your system, you can optionally install RHCOS using IBM Secure Execution before proceeding to the fast-track installation.

19.2.12.1. Installing RHCOS using IBM Secure Execution

Before you install RHCOS using IBM Secure Execution, you must prepare the underlying infrastructure.



IMPORTANT

Installing RHCOS using IBM Secure Execution is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

Prerequisites

- IBM z15 or later, or IBM® LinuxONE III or later.
- Red Hat Enterprise Linux (RHEL) 8 or later.
- You have a bootstrap Ignition file. The file is not protected, enabling others to view and edit it.

- You have verified that the boot image has not been altered after installation.
- You must run all your nodes as IBM Secure Execution guests.

Procedure

1. Prepare your RHEL KVM host to support IBM Secure Execution.

- By default, KVM hosts do not support guests in IBM Secure Execution mode. To support guests in IBM Secure Execution mode, KVM hosts must boot in LPAR mode with the kernel parameter specification **prot_virt=1**. To enable **prot_virt=1** on RHEL 8, follow these steps:
 - Navigate to **/boot/loader/entries/** to modify your bootloader configuration file ***.conf**.
 - Add the kernel command line parameter **prot_virt=1**.
 - Run the **zipl** command and reboot your system.
KVM hosts that successfully start with support for IBM Secure Execution for Linux issue the following kernel message:

```
prot_virt: Reserving <amount>MB as ultravisor base storage.
```

- To verify that the KVM host now supports IBM Secure Execution, run the following command:

```
# cat /sys/firmware/uv/prot_virt_host
```

Example output

```
1
```

The value of this attribute is 1 for Linux instances that detect their environment as consistent with that of a secure host. For other instances, the value is 0.

2. Add your host keys to the KVM guest via Ignition.

During the first boot, RHCOS looks for your host keys to re-encrypt itself with them. RHCOS searches for files starting with **ibm-z-hostkey-** in the **/etc/se-hostkeys** directory. All host keys, for each machine the cluster is running on, must be loaded into the directory by the administrator. After first boot, you cannot run the VM on any other machines.



NOTE

You need to prepare your Ignition file on a safe system. For example, another IBM Secure Execution guest.

For example:

```
{
  "ignition": { "version": "3.0.0" },
  "storage": {
    "files": [
      {
        "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
        "contents": {
```

```

    "source": "data:;base64,<base64 encoded hostkey document>"
  },
  "mode": 420
},
{
  "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
  "contents": {
    "source": "data:;base64,<base64 encoded hostkey document>"
  },
  "mode": 420
}
]
}
}
...

```



NOTE

You can add as many host keys as required if you want your node to be able to run on multiple IBM Z machines.

- To generate the Base64 encoded string, run the following command:

```
base64 <your-hostkey>.crt
```

Compared to guests not running IBM Secure Execution, the first boot of the machine is longer because the entire image is encrypted with a randomly generated LUKS passphrase before the Ignition phase.

- Follow the fast-track installation procedure to install nodes using the IBM Secure Execution QCOW image.

Additional resources

- [Introducing IBM Secure Execution for Linux](#)
- [Linux as an IBM Secure Execution host or guest](#)

19.2.12.2. Fast-track installation by using a prepackaged QCOW2 disk image

Complete the following steps to create the machines in a fast-track installation of Red Hat Enterprise Linux CoreOS (RHCOS), importing a prepackaged Red Hat Enterprise Linux CoreOS (RHCOS) QEMU copy-on-write (QCOW2) disk image.

Prerequisites

- At least one LPAR running on RHEL 8.4 or later with KVM, referred to as RHEL KVM host in this procedure.
- The KVM/QEMU hypervisor is installed on the RHEL KVM host.
- A domain name server (DNS) that can perform hostname and reverse lookup for the nodes.
- A DHCP server that provides IP addresses.

Procedure

1. Obtain the RHEL QEMU copy-on-write (QCOW2) disk image file from the [Product Downloads](#) page on the Red Hat Customer Portal or from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate RHCOS QCOW2 image described in the following procedure.

2. Download the QCOW2 disk image and Ignition files to a common directory on the RHEL KVM host.

For example: **`/var/lib/libvirt/images`**



NOTE

The Ignition files are generated by the OpenShift Container Platform installer.

3. Create a new disk image with the QCOW2 disk image backing file for each KVM guest node.

```
$ qemu-img create -f qcow2 -F qcow2 -b /var/lib/libvirt/images/{source_rhcos_qemu}
/var/lib/libvirt/images/{vmname}.qcow2 {size}
```

4. Create the new KVM guest nodes using the Ignition file and the new disk image.

```
$ virt-install --noautoconsole \
  --connect qemu:///system \
  --name {vn_name} \
  --memory {memory} \
  --vcpus {vcpus} \
  --disk {disk} \
  --import \
  --network network={network},mac={mac} \
  --disk path={ign_file},format=raw,readonly=on,serial=ignition,startup_policy=optional
```

19.2.12.3. Full installation on a new QCOW2 disk image

Complete the following steps to create the machines in a full installation on a new QEMU copy-on-write (QCOW2) disk image.

Prerequisites

- At least one LPAR running on RHEL 8.4 or later with KVM, referred to as RHEL KVM host in this procedure.
- The KVM/QEMU hypervisor is installed on the RHEL KVM host.
- A domain name server (DNS) that can perform hostname and reverse lookup for the nodes.
- An HTTP or HTTPS server is set up.

Procedure

1. Obtain the RHEL kernel, initramfs, and rootfs files from the [Product Downloads](#) page on the Red Hat Customer Portal or from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate RHCOS QCOW2 image described in the following procedure.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- kernel: **rhcos-`<version>`-live-kernel-`<architecture>`**
 - initramfs: **rhcos-`<version>`-live-initramfs.`<architecture>`.img**
 - rootfs: **rhcos-`<version>`-live-rootfs.`<architecture>`.img**
2. Move the downloaded RHEL live kernel, initramfs, and rootfs as well as the Ignition files to an HTTP or HTTPS server before you launch **virt-install**.



NOTE

The Ignition files are generated by the OpenShift Container Platform installer.

3. Create the new KVM guest nodes using the RHEL kernel, initramfs, and ignition files, the new disk image, and adjusted parm line arguments.
 - For **--location**, specify the location of the kernel/initrd on the HTTP or HTTPS server.
 - For **coreos.inst.ignition_url=**, specify the Ignition file for the machine role. Use **bootstrap.ign**, **master.ign**, or **worker.ign**. Only HTTP and HTTPS protocols are supported.
 - For **coreos.live.rootfs_url=**, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.

```
$ virt-install \
  --connect qemu:///system \
  --name {vn_name} \
  --vcpus {vcpus} \
  --memory {memory_mb} \
  --disk {vn_name}.qcow2,size={image_size| default(10,true)} \
  --network network={virt_network_parm} \
  --boot hd \
  --location {media_location},kernel={rhcos_kernel},initrd={rhcos_initrd} \
  --extra-args "rd.neednet=1 coreos.inst=yes coreos.inst.install_dev=vda
  coreos.live.rootfs_url={rhcos_liveos} ip={ip}::{default_gateway}:{subnet_mask_length}:
  {vn_name}:enc1:none:{MTU} nameserver={dns} coreos.inst.ignition_url={rhcos_ign}" \
  --noautoconsole \
  --wait
```

19.2.12.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

19.2.12.4.1. Networking options for ISO installations

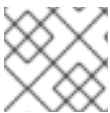
If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=** and **nameserver=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=** and **nameserver=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the **dracut.cmdline** manual page.

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```


**NOTE**

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.

**NOTE**

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

■

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none  
ip=:::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp  
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none  
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp  
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1  
nameserver=8.8.8.8
```

19.2.13. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

19.2.14. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

19.2.15. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.25.0
master-1	Ready	master	63m	v1.25.0
master-2	Ready	master	64m	v1.25.0

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-mddf5	20m	system:node:master-01.example.com	Approved,Issued
csr-z5rln	16m	system:node:worker-21.example.com	Approved,Issued

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

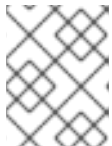
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

19.2.16. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

- Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

19.2.16.1. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

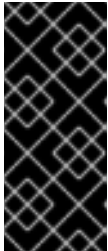
Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

19.2.16.1.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Z.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```


Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

```
NAME          VERSION          AVAILABLE PROGRESSING DEGRADED SINCE
MESSAGE
image-registry 4.12             True      False      False      6h50m
```

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

19.2.16.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

19.2.17. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE          NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running    1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8    1/1
Running    0    5m
...
```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1 Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

19.2.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service
- [How to generate SOSREPORT within OpenShift4 nodes without SSH](#) .

19.2.19. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .

19.3. INSTALLING A CLUSTER WITH RHEL KVM ON IBM Z AND IBM(R) LINUXONE IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Z or IBM® LinuxONE infrastructure that you provision in a restricted network.



NOTE

While this document refers to only IBM Z, all information in it also applies to IBM® LinuxONE.

**IMPORTANT**

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

19.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.
- You must move or remove any existing installation files, before you begin the installation process. This ensures that the required installation files are created and updated during the installation process.

**IMPORTANT**

Ensure that installation steps are done from a machine with access to the installation media.

- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

- You provisioned a RHEL Kernel Virtual Machine (KVM) system that is hosted on the logical partition (LPAR) and based on RHEL 8.4 or later. See [Red Hat Enterprise Linux 8 and 9 Life Cycle](#).

19.3.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror

host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

19.3.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

19.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

19.3.4. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

One or more KVM host machines based on RHEL 8.4 or later. Each RHEL KVM host machine must have libvirt installed and running. The virtual machines are provisioned under each RHEL KVM host machine.

19.3.4.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

Table 19.18. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.

**IMPORTANT**

To improve high availability of your cluster, distribute the control plane machines over different RHEL instances on at least two physical machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

See [Red Hat Enterprise Linux technology capabilities and limits](#) .

19.3.4.2. Network connectivity requirements

The OpenShift Container Platform installer creates the Ignition files, which are necessary for all the Red Hat Enterprise Linux CoreOS (RHCOS) virtual machines. The automated installation of OpenShift Container Platform is performed by the bootstrap machine. It starts the installation of OpenShift Container Platform on each node, starts the Kubernetes cluster, and then finishes. During this bootstrap, the virtual machine must have an established network connection either through a Dynamic Host Configuration Protocol (DHCP) server or static IP address.

19.3.4.3. IBM Z network connectivity requirements

To install on IBM Z under RHEL KVM, you need:

- A RHEL KVM host configured with an OSA or RoCE network adapter.
- Either a RHEL KVM host that is configured to use bridged networking in libvirt or MacVTap to connect the network to the guests.

See [Types of virtual network connections](#) .

19.3.4.4. Host machine resource requirements

The RHEL KVM host in your environment must meet the following requirements to host the virtual machines that you plan for the OpenShift Container Platform environment. See [Getting started with virtualization](#).

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM z16 (all models), IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- IBM® LinuxONE Emperor 4, IBM® LinuxONE III (all models), IBM® LinuxONE Emperor II, IBM® LinuxONE Rockhopper II, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper

**NOTE**

Support for RHCOS functionality for IBM z13 all models, IBM® LinuxONE Emperor, and IBM® LinuxONE Rockhopper is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

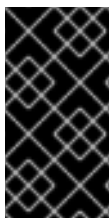
19.3.4.5. Minimum IBM Z system environment

Hardware requirements

- The equivalent of six Integrated Facilities for Linux (IFL), which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.

**NOTE**

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.

**IMPORTANT**

Since the overall performance of the cluster can be impacted, the LPARs that are used to set up the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

Operating system requirements

- One LPAR running on RHEL 8.4 or later with KVM, which is managed by libvirt

On your RHEL KVM host, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

19.3.4.6. Minimum resource requirements

Each cluster virtual machine must meet the following minimum requirements:

Virtual Machine	Operating System	vCPU [1]	Virtual RAM	Storage	IOPS
Bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. One physical core (IFL) provides two logical cores (threads) when SMT-2 is enabled. The hypervisor can provide two or more vCPUs.

19.3.4.7. Preferred IBM Z system environment

Hardware requirements

- Three LPARS that each have the equivalent of six IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.

Operating system requirements

- For high availability, two or three LPARs running on RHEL 8.4 or later with KVM, which are managed by libvirt.

On your RHEL KVM host, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines, distributed across the RHEL KVM host machines.
- At least six guest virtual machines for OpenShift Container Platform compute machines, distributed across the RHEL KVM host machines.
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
- To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using **cpu_shares**. Do the same for infrastructure nodes, if they exist. See [schedinfo](#) in IBM Documentation.

19.3.4.8. Preferred resource requirements

The preferred requirements for each cluster virtual machine are:

Virtual Machine	Operating System	vCPU	Virtual RAM	Storage
Bootstrap	RHCOS	4	16 GB	120 GB
Control plane	RHCOS	8	16 GB	120 GB
Compute	RHCOS	6	8 GB	120 GB

Virtual Machine	Operating System	vCPU	Virtual RAM	Storage
-----------------	------------------	------	-------------	---------

19.3.4.9. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

Additional resources

- [Recommended host practices for IBM Z & IBM® LinuxONE environments](#)

19.3.4.10. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

19.3.4.10.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a

reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

19.3.4.10.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

Table 19.19. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 19.20. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 19.21. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

19.3.4.11. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:


- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 19.22. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

19.3.4.11.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 19.4. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.

- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

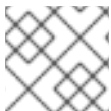
Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 19.5. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

19.3.4.12. Load balancing requirements for user-provisioned infrastructure

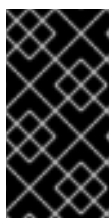
Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 19.23. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 19.24. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

19.3.4.12.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 19.6. Sample API and application Ingress load balancer configuration

```
global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request  10s
  timeout queue      1m
  timeout connect    10s
  timeout client     1m
```

```

timeout server      1m
timeout http-keep-alive 10s
timeout check      10s
maxconn            3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

19.3.5. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.

**NOTE**

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Choose to perform either a fast track installation of Red Hat Enterprise Linux CoreOS (RHCOS) or a full installation of Red Hat Enterprise Linux CoreOS (RHCOS). For the full installation, you must set up an HTTP or HTTPS server to provide Ignition files and install images to the cluster nodes. For the fast track installation an HTTP or HTTPS server is not required, however, a DHCP server is required. See sections "Fast-track installation: Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines" and "Full installation: Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines".
3. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
4. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.

**IMPORTANT**

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

5. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
6. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.

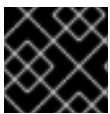
7. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

19.3.6. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1 Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

- 1 Provides the record name for the Kubernetes internal API.
- 2 Provides the record name for the Kubernetes API.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

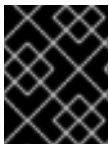
- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

19.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```


- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

19.3.8. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

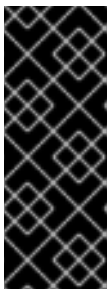
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

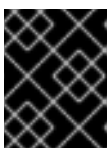
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

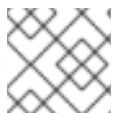


IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

19.3.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

19.3.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 19.25. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

19.3.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.

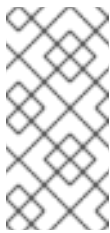

```
networking:
  clusterNetwork:
```


- cidr: 10.128.0.0/14
hostPrefix: 23
- cidr: fd00:10:128::/56
hostPrefix: 64
- serviceNetwork:
 - 172.30.0.0/16
 - fd00:172:16::/112

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 19.26. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.network Type	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example:  networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .

Parameter	Description	Values
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


19.3.8.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 19.27. Optional parameters

Parameter	Description	Values
-----------	-------------	--------



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String

Parameter	Description	Values
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p>  <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.

Parameter	Description	Values
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are s390x (the default).	String
controlPlane.hypertreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

19.3.8.2. Sample install-config.yaml file for IBM Z

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker

```

```

replicas: 0 4
architecture: s390x
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
  architecture: s390x
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths":{"<local_registry>":{"auth": "<credentials>","email": "you@example.com"}}}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
additionalTrustBundle: | 17
-----BEGIN CERTIFICATE-----
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
-----END CERTIFICATE-----
imageContentSources: 18
- mirrors:
- <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
- <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not available on your OpenShift Container Platform nodes, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether on your OpenShift Container Platform nodes or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Z infrastructure.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 17 Add the **additionalTrustBundle** parameter and value. The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.
- 18 Provide the **imageContentSources** section from the output of the command to mirror the repository.

19.3.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of

them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

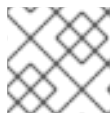
For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$. /openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

19.3.8.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a minimal three node cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

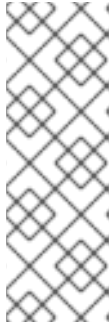
Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

**NOTE**

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

**NOTE**

The preferred resource for control plane nodes is six vCPUs and 21 GB. For three control plane nodes this is the memory + vCPU equivalent of a minimum five-node cluster. You should back the three nodes, each installed on a 120 GB disk, with three IFLs that are SMT2 enabled. The minimum tested setup is three vCPUs and 10 GB on a 120 GB disk for each control plane node.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

19.3.9. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

19.3.9.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 19.28. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 19.29. defaultNetwork object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 19.30. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>

Field	Type	Description
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 19.31. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.

Field	Type	Description
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <p> NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is 2⁽²³⁻¹⁴⁾=512.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 19.32. `policyAuditConfig` object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 19.33. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 19.34. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

19.3.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.



NOTE

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program runs on s390x only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.



WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.



IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

19.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Z infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) as Red Hat Enterprise Linux (RHEL) guest virtual machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

You can perform a fast-track installation of RHCOS that uses a prepackaged QEMU copy-on-write (QCOW2) disk image. Alternatively, you can perform a full installation on a new QCOW2 disk image.

To add further security to your system, you can optionally install RHCOS using IBM Secure Execution before proceeding to the fast-track installation.

19.3.11.1. Installing RHCOS using IBM Secure Execution

Before you install RHCOS using IBM Secure Execution, you must prepare the underlying infrastructure.



IMPORTANT

Installing RHCOS using IBM Secure Execution is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

Prerequisites

- IBM z15 or later, or IBM® LinuxONE III or later.

- Red Hat Enterprise Linux (RHEL) 8 or later.
- You have a bootstrap Ignition file. The file is not protected, enabling others to view and edit it.
- You have verified that the boot image has not been altered after installation.
- You must run all your nodes as IBM Secure Execution guests.

Procedure

1. Prepare your RHEL KVM host to support IBM Secure Execution.

- By default, KVM hosts do not support guests in IBM Secure Execution mode. To support guests in IBM Secure Execution mode, KVM hosts must boot in LPAR mode with the kernel parameter specification **prot_virt=1**. To enable **prot_virt=1** on RHEL 8, follow these steps:
 - a. Navigate to **/boot/loader/entries/** to modify your bootloader configuration file ***.conf**.
 - b. Add the kernel command line parameter **prot_virt=1**.
 - c. Run the **zipl** command and reboot your system.
KVM hosts that successfully start with support for IBM Secure Execution for Linux issue the following kernel message:

```
prot_virt: Reserving <amount>MB as ultravisor base storage.
```

- d. To verify that the KVM host now supports IBM Secure Execution, run the following command:

```
# cat /sys/firmware/uv/prot_virt_host
```

Example output

```
1
```

The value of this attribute is 1 for Linux instances that detect their environment as consistent with that of a secure host. For other instances, the value is 0.

2. Add your host keys to the KVM guest via Ignition.

During the first boot, RHCOS looks for your host keys to re-encrypt itself with them. RHCOS searches for files starting with **ibm-z-hostkey-** in the **/etc/se-hostkeys** directory. All host keys, for each machine the cluster is running on, must be loaded into the directory by the administrator. After first boot, you cannot run the VM on any other machines.



NOTE

You need to prepare your Ignition file on a safe system. For example, another IBM Secure Execution guest.

For example:

```
{
  "ignition": { "version": "3.0.0" },
  "storage": {
```

```

"files": [
  {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
      "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
  },
  {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
      "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
  }
]
}
...

```



NOTE

You can add as many host keys as required if you want your node to be able to run on multiple IBM Z machines.

- To generate the Base64 encoded string, run the following command:

```
base64 <your-hostkey>.crt
```

Compared to guests not running IBM Secure Execution, the first boot of the machine is longer because the entire image is encrypted with a randomly generated LUKS passphrase before the Ignition phase.

- Follow the fast-track installation procedure to install nodes using the IBM Secure Execution QCOW image.

Additional resources

- [Introducing IBM Secure Execution for Linux](#)
- [Linux as an IBM Secure Execution host or guest](#)

19.3.11.2. Fast-track installation by using a prepackaged QCOW2 disk image

Complete the following steps to create the machines in a fast-track installation of Red Hat Enterprise Linux CoreOS (RHCOS), importing a prepackaged Red Hat Enterprise Linux CoreOS (RHCOS) QEMU copy-on-write (QCOW2) disk image.

Prerequisites

- At least one LPAR running on RHEL 8.4 or later with KVM, referred to as RHEL KVM host in this procedure.
- The KVM/QEMU hypervisor is installed on the RHEL KVM host.

- A domain name server (DNS) that can perform hostname and reverse lookup for the nodes.
- A DHCP server that provides IP addresses.

Procedure

1. Obtain the RHEL QEMU copy-on-write (QCOW2) disk image file from the [Product Downloads](#) page on the Red Hat Customer Portal or from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate RHCOS QCOW2 image described in the following procedure.

2. Download the QCOW2 disk image and Ignition files to a common directory on the RHEL KVM host.

For example: `/var/lib/libvirt/images`



NOTE

The Ignition files are generated by the OpenShift Container Platform installer.

3. Create a new disk image with the QCOW2 disk image backing file for each KVM guest node.

```
$ qemu-img create -f qcow2 -F qcow2 -b /var/lib/libvirt/images/{source_rhcos_qemu}
/var/lib/libvirt/images/{vmname}.qcow2 {size}
```

4. Create the new KVM guest nodes using the Ignition file and the new disk image.

```
$ virt-install --noautoconsole \
  --connect qemu:///system \
  --name {vn_name} \
  --memory {memory} \
  --vcpus {vcpus} \
  --disk {disk} \
  --import \
  --network network={network},mac={mac} \
  --disk path={ign_file},format=raw,readonly=on,serial=ignition,startup_policy=optional
```

19.3.11.3. Full installation on a new QCOW2 disk image

Complete the following steps to create the machines in a full installation on a new QEMU copy-on-write (QCOW2) disk image.

Prerequisites

- At least one LPAR running on RHEL 8.4 or later with KVM, referred to as RHEL KVM host in this procedure.
- The KVM/QEMU hypervisor is installed on the RHEL KVM host.

- A domain name server (DNS) that can perform hostname and reverse lookup for the nodes.
- An HTTP or HTTPS server is set up.

Procedure

1. Obtain the RHEL kernel, initramfs, and rootfs files from the [Product Downloads](#) page on the Red Hat Customer Portal or from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate RHCOS QCOW2 image described in the following procedure.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- kernel: **rhcos-`<version>`-live-kernel-`<architecture>`**
 - initramfs: **rhcos-`<version>`-live-initramfs.`<architecture>`.img**
 - rootfs: **rhcos-`<version>`-live-rootfs.`<architecture>`.img**
2. Move the downloaded RHEL live kernel, initramfs, and rootfs as well as the Ignition files to an HTTP or HTTPS server before you launch **virt-install**.



NOTE

The Ignition files are generated by the OpenShift Container Platform installer.

3. Create the new KVM guest nodes using the RHEL kernel, initramfs, and Ignition files, the new disk image, and adjusted parm line arguments.
 - For **--location**, specify the location of the kernel/initrd on the HTTP or HTTPS server.
 - For **coreos.inst.ignition_url=**, specify the Ignition file for the machine role. Use **bootstrap.ign**, **master.ign**, or **worker.ign**. Only HTTP and HTTPS protocols are supported.
 - For **coreos.live.rootfs_url=**, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.

```
$ virt-install \
  --connect qemu:///system \
  --name {vn_name} \
  --vcpus {vcpus} \
  --memory {memory_mb} \
  --disk {vn_name}.qcow2,size={image_size| default(10,true)} \
  --network network={virt_network_parm} \
  --boot hd \
  --location {media_location},kernel={rhcos_kernel},initrd={rhcos_initrd} \
  --extra-args "rd.neednet=1 coreos.inst=yes coreos.inst.install_dev=vda
  coreos.live.rootfs_url={rhcos_liveos} ip={ip}::{default_gateway}:{subnet_mask_length}:
```



```
{vn_name}:enc1:none:{MTU} nameserver={dns} coreos.inst.ignition_url={rhcos_ign}" \
--noautoconsole \
--wait
```

19.3.11.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

19.3.11.4.1. Networking options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=** and **nameserver=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=** and **nameserver=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the **dracut.cmdline** manual page.

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

**NOTE**

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.

**NOTE**

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

■

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=:::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

19.3.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

19.3.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

19.3.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.25.0
master-1	Ready	master	63m	v1.25.0
master-2	Ready	master	64m	v1.25.0

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-8b2br	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
csr-8vnps	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
...			

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
```

```
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

19.3.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

19.3.15.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:


```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

19.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

19.3.15.2.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Z.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.

**NOTE**

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```

**NOTE**

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

19.3.15.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

19.3.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

- Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.12.0	True	False	False

baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running  1      9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8  1/1
Running  0      5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

4. Register your cluster on the [Cluster registration](#) page.

Additional resources

- [How to generate SOSREPORT within OpenShift Container Platform version 4 nodes without SSH.](#)

19.3.17. Next steps

- [Customize your cluster.](#)
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)

CHAPTER 20. INSTALLING ON IBM POWER

20.1. PREPARING TO INSTALL ON IBM POWER

20.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

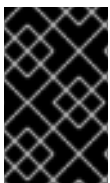
20.1.2. Choosing a method to install OpenShift Container Platform on IBM Power

You can install a cluster on IBM Power infrastructure that you provision, by using one of the following methods:

- **Installing a cluster on IBM Power** You can install OpenShift Container Platform on IBM Power infrastructure that you provision.
- **Installing a cluster on IBM Power in a restricted network** You can install OpenShift Container Platform on IBM Power infrastructure that you provision in a restricted or disconnected network, by using an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

20.2. INSTALLING A CLUSTER ON IBM POWER

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Power infrastructure that you provision.



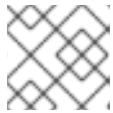
IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

20.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- Before you begin the installation process, you must clean the installation directory. This ensures that the required installation files are created and updated during the installation process.
- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

20.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

20.2.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

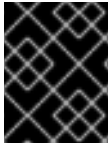
20.2.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 20.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.

Hosts	Description
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

20.2.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 20.2. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

20.2.3.3. Minimum IBM Power requirements

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM Power9 or Power10 processor-based systems

**NOTE**

Support for RHCOS functionality for all IBM POWER8 models, IBM POWER9 AC922, IBM POWER9 IC922, and IBM POWER9 LC922 is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

Hardware requirements

- Six IBM Power bare metal servers or six LPARs across multiple PowerVM servers

Operating system requirements

- One instance of an IBM Power9 or Power10 processor-based system

On your IBM Power instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

Disk storage for the IBM Power guest virtual machines

- Local storage, or storage provisioned by the Virtual I/O Server using vSCSI, NPIV (N-Port ID Virtualization) or SSP (shared storage pools)

Network for the PowerVM guest virtual machines

- Dedicated physical adapter, or SR-IOV virtual function
- Available by the Virtual I/O Server using Shared Ethernet Adapter
- Virtualized by the Virtual I/O Server using IBM vNIC

Storage / main memory

- 100 GB / 16 GB for OpenShift Container Platform control plane machines
- 100 GB / 8 GB for OpenShift Container Platform compute machines
- 100 GB / 16 GB for the temporary OpenShift Container Platform bootstrap machine

20.2.3.4. Recommended IBM Power system requirements**Hardware requirements**

- Six IBM Power bare metal servers or six LPARs across multiple PowerVM servers

Operating system requirements

- One instance of an IBM Power9 or Power10 processor-based system

On your IBM Power instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

Disk storage for the IBM Power guest virtual machines

- Local storage, or storage provisioned by the Virtual I/O Server using vSCSI, NPIV (N-Port ID Virtualization) or SSP (shared storage pools)

Network for the PowerVM guest virtual machines

- Dedicated physical adapter, or SR-IOV virtual function
- Available by the Virtual I/O Server using Shared Ethernet Adapter
- Virtualized by the Virtual I/O Server using IBM vNIC

Storage / main memory

- 120 GB / 32 GB for OpenShift Container Platform control plane machines
- 120 GB / 32 GB for OpenShift Container Platform compute machines
- 120 GB / 16 GB for the temporary OpenShift Container Platform bootstrap machine

20.2.3.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

20.2.3.6. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.

**NOTE**

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

20.2.3.6.1. Setting the cluster node hostnames through DHCP

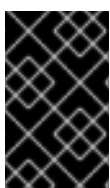
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

20.2.3.6.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

**IMPORTANT**

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 20.3. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves

Protocol	Port	Description
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 20.4. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 20.5. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

20.2.3.7. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

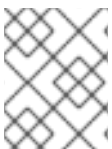
It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 20.6. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster. For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

20.2.3.7.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 20.1. Sample DNS zone database

```

$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 20.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

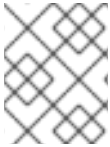
- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

20.2.3.8. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 20.7. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 20.8. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

20.2.3.8.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 20.3. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode                http
  log                 global
  option              dontlognull
  option http-server-close
  option              redispatch
  retries              3
  timeout http-request 10s
  timeout queue        1m
  timeout connect      10s
  timeout client        1m
  timeout server        1m
  timeout http-keep-alive 10s
  timeout check         10s
  maxconn              3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5

```

```

bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltupe** on the HAProxy node.

20.2.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.

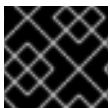
- a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

20.2.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```


- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

20.2.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

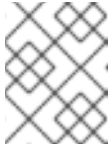
- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
-
```

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

20.2.7. Obtaining the installation program

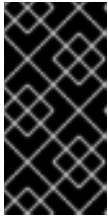
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

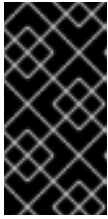
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

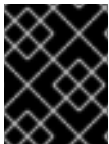
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

20.2.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.

To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

20.2.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

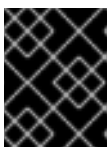
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

20.2.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

20.2.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 20.9. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

20.2.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.

```
networking:
  clusterNetwork:
```


- cidr: 10.128.0.0/14
hostPrefix: 23
- cidr: fd00:10:128::/56
hostPrefix: 64
- serviceNetwork:
 - 172.30.0.0/16
 - fd00:172:16::/112

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 20.10. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>
networking.network Type	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	<p>The IP address blocks for pods.</p> <p>The default value is 10.128.0.0/14 with a host prefix of /23.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre style="margin-left: 20px;">networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	<p>Required if you use networking.clusterNetwork. An IP address block.</p> <p>An IPv4 network.</p>	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .

Parameter	Description	Values
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


20.2.9.1.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 20.11. Optional parameters

Parameter	Description	Values
-----------	-------------	--------



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are ppc64le (the default).	String

Parameter	Description	Values
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.

Parameter	Description	Values
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are ppc64le (the default).	String
controlPlane.hypertreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 592 1391" style="background-color: black; width: 66px; height: 359px; margin-bottom: 10px;"></div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="486 1440 592 1637" style="background-color: black; width: 66px; height: 88px; margin-bottom: 10px;"></div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;">  </div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;">  </div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

20.2.9.2. Sample install-config.yaml file for IBM Power

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker

```



```

replicas: 0 4
architecture: ppc64le
controlPlane: 5
hyperthreading: Enabled 6
name: master
replicas: 3 7
architecture: ppc64le
metadata:
  name: test 8
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14 9
      hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
    - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.

**NOTE**

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

**NOTE**

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Power infrastructure.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

20.2.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
```

```

proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

20.2.9.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

**NOTE**

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

20.2.10. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

20.2.10.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 20.12. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 20.13. defaultNetwork object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 20.14. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 20.15. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 20.16. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 20.17. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 20.18. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 40px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, #ccc 2px, #ccc 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 10px; height: 100px; background-color: #ccc; margin-right: 10px;"></div> <div> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre> </div> </div>

20.2.11. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**NOTE**

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program (without an architecture postfix) runs on ppc64le only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

**IMPORTANT**

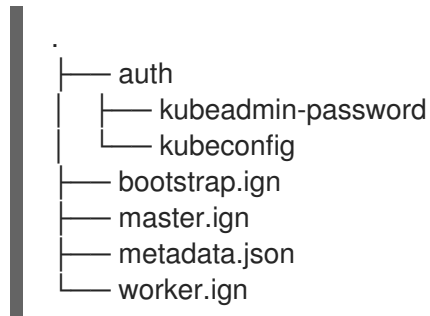
When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:



20.2.12. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Power infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Follow either the steps to use an ISO image or network PXE booting to install RHCOS on the machines.

20.2.12.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:
 - Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.

**NOTE**

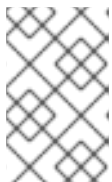
It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

1 1 You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.

2 The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.

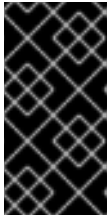
**NOTE**

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the **/dev/sda** device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

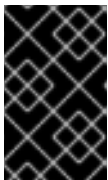
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

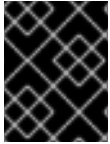
RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

20.2.12.1.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

20.2.12.1.1.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**

- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option.

Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

20.2.12.2. Installing RHCOS by using PXE booting

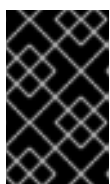
You can use PXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

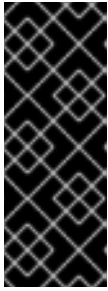
1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:



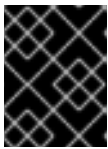
IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** `rhcos-<version>-live-kernel-<architecture>`
- **initramfs:** `rhcos-<version>-live-initramfs.<architecture>.img`
- **rootfs:** `rhcos-<version>-live-rootfs.<architecture>.img`

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.
6. Configure PXE installation for the RHCOS images and begin the installation. Modify the following example menu entry for your environment and verify that the image and Ignition files are properly accessible:

```

DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.<architecture>.img
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
  coreos.inst.install_dev=/dev/sda coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
2 3

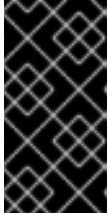
```

- 1** Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2** If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3** Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

7. Monitor the progress of the RHCOS installation on the console of the machine.

**IMPORTANT**

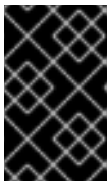
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.
9. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

20.2.12.3. Enabling multipathing with kernel arguments on RHCOS

In OpenShift Container Platform 4.9 or later, during installation, you can enable multipathing for provisioned nodes. RHCOS supports multipathing on the primary disk. Multipathing provides added benefits of stronger resilience to hardware failure to achieve higher host availability.

During the initial cluster creation, you might want to add kernel arguments to all master or worker nodes. To add kernel arguments to master or worker nodes, you can create a **MachineConfig** object and inject that object into the set of manifest files used by Ignition during cluster setup.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. Decide if you want to add kernel arguments to worker or control plane nodes.
 - Create a machine config file. For example, create a **99-master-kargs-mpath.yaml** that instructs the cluster to add the **master** label and identify the multipath kernel argument:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "master"
  name: 99-master-kargs-mpath
spec:
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

3. To enable multipathing on worker nodes:
 - Create a machine config file. For example, create a **99-worker-kargs-mpath.yaml** that instructs the cluster to add the **worker** label and identify the multipath kernel argument:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
```

```

machineconfiguration.openshift.io/role: "worker"
name: 99-worker-kargs-mpath
spec:
  kernelArguments:
  - 'rd.multipath=default'
  - 'root=/dev/disk/by-label/dm-mpath-root'

```

You can now continue on to create the cluster.



IMPORTANT

Additional postinstallation steps are required to fully enable multipathing. For more information, see “Enabling multipathing with kernel arguments on RHCOS” in *Postinstallation machine configuration tasks*.

In case of MPIO failure, use the `bootlist` command to update the boot device list with alternate logical device names. The command displays a boot list and it designates the possible boot devices for when the system is booted in normal mode.

- a. To display a boot list and specify the possible boot devices if the system is booted in normal mode, enter the following command:

```

$ bootlist -m normal -o
sda

```

- b. To update the boot list for normal mode and add alternate device names, enter the following command:

```

$ bootlist -m normal -o /dev/sdc /dev/sdd /dev/sde
sdc
sdd
sde

```

If the original boot disk path is down, the node reboots from the alternate device registered in the normal boot device list.

20.2.13. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

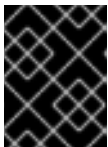
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

20.2.14. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

20.2.15. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

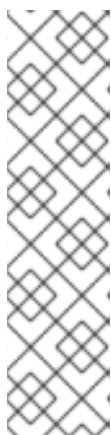
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```

NAME      AGE    REQUESTOR                                CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...

```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

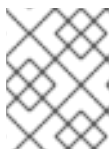
```
$ oc get nodes
```

Example output

```

NAME      STATUS    ROLES    AGE    VERSION
master-0  Ready    master   73m    v1.25.0
master-1  Ready    master   73m    v1.25.0
master-2  Ready    master   74m    v1.25.0
worker-0  Ready    worker   11m    v1.25.0
worker-1  Ready    worker   11m    v1.25.0

```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

20.2.16. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

20.2.16.1. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

20.2.16.1.1. Configuring registry storage for IBM Power

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Power.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

20.2.16.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

20.2.17. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m

```
operator-lifecycle-manager      4.12.0 True    False   False   37m
operator-lifecycle-manager-catalog 4.12.0 True    False   False   37m
operator-lifecycle-manager-packageserver 4.12.0 True    False   False   32m
service-ca                     4.12.0 True    False   False   38m
storage                         4.12.0 True    False   False   37m
```

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```
NAMESPACE          NAME                                     READY STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running    1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
```

```

1m
openshift-apiserver          apiserver-z25h4           1/1   Running   0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8 1/1
Running   0       5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1 Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Additional steps are required to enable multipathing. Do not enable multipathing during installation.

See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

20.2.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

20.2.19. Next steps

- [Enabling multipathing with kernel arguments on RHCOS](#) .
- [Customize your cluster](#) .
- If necessary, you can [opt out of remote health reporting](#) .

20.3. INSTALLING A CLUSTER ON IBM POWER IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a cluster on IBM Power infrastructure that you provision in a restricted network.

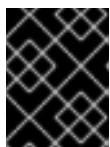


IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you install an OpenShift Container Platform cluster.

20.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a mirror registry for installation in a restricted network](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.
- Before you begin the installation process, you must move or remove any existing installation files. This ensures that the required installation files are created and updated during the installation process.



IMPORTANT

Ensure that installation steps are performed on a machine with access to the installation media.

- You provisioned [persistent storage using OpenShift Data Foundation](#) or other supported storage protocols for your cluster. To deploy a private image registry, you must set up persistent storage with **ReadWriteMany** access.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



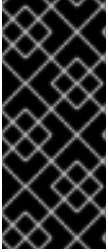
NOTE

Be sure to also review this site list if you are configuring a proxy.

20.3.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

20.3.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

20.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

20.3.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

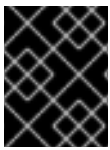
This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

20.3.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 20.19. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap, control plane, and compute machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#) .

20.3.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 20.20. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

20.3.4.3. Minimum IBM Power requirements

You can install OpenShift Container Platform version 4.12 on the following IBM hardware:

- IBM Power9 or Power10 processor-based systems



NOTE

Support for RHCOS functionality for all IBM POWER8 models, IBM POWER9 AC922, IBM POWER9 IC922, and IBM POWER9 LC922 is deprecated. These hardware models remain fully supported in OpenShift Container Platform 4.12. However, Red Hat recommends that you use later hardware models.

Hardware requirements

- Six IBM Power bare metal servers or six LPARs across multiple PowerVM servers

Operating system requirements

- One instance of an IBM Power9 or Power10 processor-based system

On your IBM Power instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

Disk storage for the IBM Power guest virtual machines

- Local storage, or storage provisioned by the Virtual I/O Server using vSCSI, NPIV (N-Port ID Virtualization) or SSP (shared storage pools)

Network for the PowerVM guest virtual machines

- Dedicated physical adapter, or SR-IOV virtual function
- Available by the Virtual I/O Server using Shared Ethernet Adapter
- Virtualized by the Virtual I/O Server using IBM vNIC

Storage / main memory

- 100 GB / 16 GB for OpenShift Container Platform control plane machines
- 100 GB / 8 GB for OpenShift Container Platform compute machines
- 100 GB / 16 GB for the temporary OpenShift Container Platform bootstrap machine

20.3.4.4. Recommended IBM Power system requirements

Hardware requirements

- Six IBM Power bare metal servers or six LPARs across multiple PowerVM servers

Operating system requirements

- One instance of an IBM Power9 or Power10 processor-based system

On your IBM Power instance, set up:

- Three guest virtual machines for OpenShift Container Platform control plane machines
- Two guest virtual machines for OpenShift Container Platform compute machines
- One guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

Disk storage for the IBM Power guest virtual machines

- Local storage, or storage provisioned by the Virtual I/O Server using vSCSI, NPIV (N-Port ID Virtualization) or SSP (shared storage pools)

Network for the PowerVM guest virtual machines

- Dedicated physical adapter, or SR-IOV virtual function
- Available by the Virtual I/O Server using Shared Ethernet Adapter
- Virtualized by the Virtual I/O Server using IBM vNIC

Storage / main memory

- 120 GB / 32 GB for OpenShift Container Platform control plane machines
- 120 GB / 32 GB for OpenShift Container Platform compute machines
- 120 GB / 16 GB for the temporary OpenShift Container Platform bootstrap machine

20.3.4.5. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

20.3.4.6. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The

Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

20.3.4.6.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

20.3.4.6.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

Table 20.21. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .

Protocol	Port	Description
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 20.22. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 20.23. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

20.3.4.7. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 20.24. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
	api-int.<cluster_name>.<base_domain>.	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

20.3.4.7.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 20.4. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines

application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 20.5. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.

- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

20.3.4.8. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

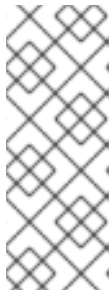
Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 20.25. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server

Port	Back-end machines (pool members)	Internal	External	Description
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 20.26. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

20.3.4.8.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 20.6. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request  10s
  timeout queue        1m
  timeout connect      10s
  timeout client       1m
  timeout server       1m
  timeout http-keep-alive 10s
  timeout check        10s
  maxconn             3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s

```

```

fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1 Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2 4 The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3 Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltupe** on the HAProxy node.

20.3.5. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.

3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.



NOTE

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

20.3.6. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.



IMPORTANT

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1 Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

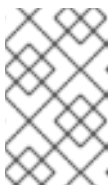
```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```


- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

20.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

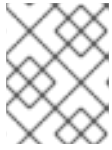
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

20.3.8. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

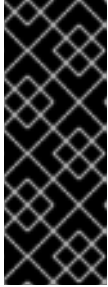
- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.

- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

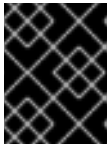
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

20.3.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

20.3.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 20.27. Required parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform.<platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

20.3.8.1.2. Network configuration parameters

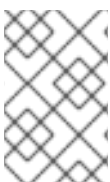
You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

- If you use the Red Hat OpenShift Networking OVN-Kubernetes network plugin, both IPv4 and IPv6 address families are supported.
- If you use the Red Hat OpenShift Networking OpenShift SDN network plugin, only the IPv4 address family is supported.

If you configure your cluster to use both IP address families, review the following requirements:

- Both IP families must use the same network interface for the default gateway.
- Both IP families must have the default gateway.
- You must specify IPv4 and IPv6 addresses in the same order for all network configuration parameters. For example, in the following configuration IPv4 addresses are listed before IPv6 addresses.


```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 20.28. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p> <p>If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


20.3.8.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 20.29. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are ppc64le (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are ppc64le (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 586 592 1391" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1442 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

20.3.8.2. Sample install-config.yaml file for IBM Power

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
  architecture : ppc64le
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7

```


**NOTE**

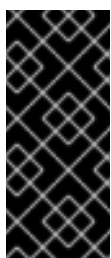
If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

**NOTE**

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for IBM Power infrastructure.

**IMPORTANT**

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 17 Provide the contents of the certificate file that you used for your mirror registry.
- 18 Provide the **imageContentSources** section from the output of the command to mirror the repository.

20.3.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

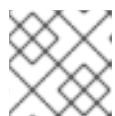
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

20.3.8.4. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



NOTE

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file is set to **true**. This enables your application workloads to run on the control plane nodes.

- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

20.3.9. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

20.3.9.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 20.30. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 20.31. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 20.32. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 20.33. ovnKubernetesConfig object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 20.34. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 20.35. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

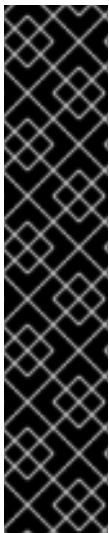
Table 20.36. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

20.3.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**NOTE**

The installation program that generates the manifest and Ignition files is architecture specific and can be obtained from the [client image mirror](#). The Linux version of the installation program (without an architecture postfix) runs on ppc64le only. This installer program is also available as a Mac OS version.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

**IMPORTANT**

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

20.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on IBM Power infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Follow either the steps to use an ISO image or network PXE booting to install RHCOS on the machines.

20.3.11.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

2. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

3. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left     Speed
  0   0   0   0   0   0   0   0   0  --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

4. Although it is possible to obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS images are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep '\.iso[^\.]'
```

Example output

```
"location": "<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



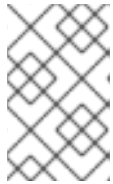
IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:
 - Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.



NOTE

It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

- 1 1 You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.

- 2 The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.



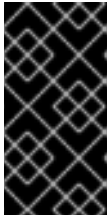
NOTE

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the **/dev/sda** device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

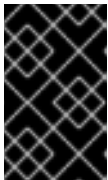
Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

20.3.11.1.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

20.3.11.1.1.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.

**IMPORTANT**

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.

**NOTE**

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

**NOTE**

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**

- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option.

Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]**
name is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

20.3.11.2. Installing RHCOS by using PXE booting

You can use PXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left     Speed
  0   0   0   0   0   0   0  --:--:--  --:--:--  --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

- Although it is possible to obtain the RHCOS **kernel**, **initramfs** and **rootfs** files that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS files are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep -Eo "https.*(kernel-|initramfs-|rootfs-)|w+(\.img)?"
```

Example output

```
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



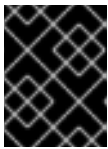
IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** `rhcos-<version>-live-kernel-<architecture>`
- **initramfs:** `rhcos-<version>-live-initramfs.<architecture>.img`
- **rootfs:** `rhcos-<version>-live-rootfs.<architecture>.img`

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.
6. Configure PXE installation for the RHCOS images and begin the installation. Modify the following example menu entry for your environment and verify that the image and Ignition files are properly accessible:

```

DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.<architecture>.img
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
  coreos.inst.install_dev=/dev/sda coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
2 3

```

- 1** **1** Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2** If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3** Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

7. Monitor the progress of the RHCOS installation on the console of the machine.

**IMPORTANT**

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.
9. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

20.3.11.3. Enabling multipathing with kernel arguments on RHCOS

In OpenShift Container Platform 4.9 or later, during installation, you can enable multipathing for provisioned nodes. RHCOS supports multipathing on the primary disk. Multipathing provides added benefits of stronger resilience to hardware failure to achieve higher host availability.

During the initial cluster creation, you might want to add kernel arguments to all master or worker nodes. To add kernel arguments to master or worker nodes, you can create a **MachineConfig** object and inject that object into the set of manifest files used by Ignition during cluster setup.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. Decide if you want to add kernel arguments to worker or control plane nodes.
 - Create a machine config file. For example, create a **99-master-kargs-mpath.yaml** that instructs the cluster to add the **master** label and identify the multipath kernel argument:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "master"
  name: 99-master-kargs-mpath
spec:
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

3. To enable multipathing on worker nodes:
 - Create a machine config file. For example, create a **99-worker-kargs-mpath.yaml** that instructs the cluster to add the **worker** label and identify the multipath kernel argument:

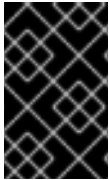
```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
```

```

machineconfiguration.openshift.io/role: "worker"
name: 99-worker-kargs-mpath
spec:
  kernelArguments:
  - 'rd.multipath=default'
  - 'root=/dev/disk/by-label/dm-mpath-root'

```

You can now continue on to create the cluster.



IMPORTANT

Additional postinstallation steps are required to fully enable multipathing. For more information, see “Enabling multipathing with kernel arguments on RHCOS” in *Postinstallation machine configuration tasks*.

In case of MPIO failure, use the `bootlist` command to update the boot device list with alternate logical device names. The command displays a boot list and it designates the possible boot devices for when the system is booted in normal mode.

- a. To display a boot list and specify the possible boot devices if the system is booted in normal mode, enter the following command:

```

$ bootlist -m normal -o
sda

```

- b. To update the boot list for normal mode and add alternate device names, enter the following command:

```

$ bootlist -m normal -o /dev/sdc /dev/sdd /dev/sde
sdc
sdd
sde

```

If the original boot disk path is down, the node reboots from the alternate device registered in the normal boot device list.

20.3.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

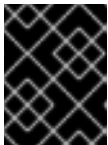
- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

20.3.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
2. Verify you can run **oc** commands successfully using the exported configuration:


```
$ oc whoami
```

Example output

```
system:admin
```

20.3.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.25.0
master-1	Ready	master	63m	v1.25.0
master-2	Ready	master	64m	v1.25.0

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

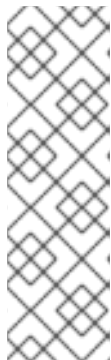
```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-8b2br	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
csr-8vnps	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
...			

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-bfd72	5m26s	system:node:ip-10-0-50-126.us-east-2.compute.internal	Pending
csr-c57lv	5m26s	system:node:ip-10-0-95-157.us-east-2.compute.internal	Pending
...			

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	73m	v1.25.0
master-1	Ready	master	73m	v1.25.0
master-2	Ready	master	74m	v1.25.0
worker-0	Ready	worker	11m	v1.25.0
worker-1	Ready	worker	11m	v1.25.0



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

20.3.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

20.3.15.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

20.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

20.3.15.2.1. Changing the image registry's management state

To start the image registry, you must change the Image Registry Operator configuration's **managementState** from **Removed** to **Managed**.

Procedure

- Change **managementState** Image Registry Operator configuration from **Removed** to **Managed**. For example:

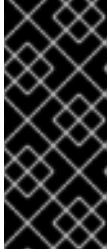
```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"managementState": "Managed"}}'
```

20.3.15.2.2. Configuring registry storage for IBM Power

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster on IBM Power.
- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

20.3.15.2.3. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

20.3.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```
NAMESPACE          NAME                                READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running    1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running    0    5m
...
```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Additional steps are required to enable multipathing. Do not enable multipathing during installation.

See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

4. Register your cluster on the [Cluster registration](#) page.

20.3.17. Next steps

- [Enabling multipathing with kernel arguments on RHCOS](#) .
- [Customize your cluster](#) .
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#) .
- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)

CHAPTER 21. INSTALLING ON OPENSTACK

21.1. PREPARING TO INSTALL ON OPENSTACK

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP).

21.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

21.1.2. Choosing a method to install OpenShift Container Platform on OpenStack

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

21.1.2.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on Red Hat OpenStack Platform (RHOSP) infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- **Installing a cluster on OpenStack with customizations** You can install a customized cluster on RHOSP. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).
- **Installing a cluster on OpenStack with Kuryr** You can install a customized OpenShift Container Platform cluster on RHOSP that uses Kuryr SDN. Kuryr and OpenShift Container Platform integration is primarily designed for OpenShift Container Platform clusters running on RHOSP VMs. Kuryr improves the network performance by plugging OpenShift Container Platform pods into RHOSP SDN. In addition, it provides interconnectivity between pods and RHOSP virtual instances.
- **Installing a cluster on OpenStack in a restricted network** You can install OpenShift Container Platform on RHOSP in a restricted or disconnected network by creating an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

21.1.2.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on RHOSP infrastructure that you provision, by using one of the following methods:

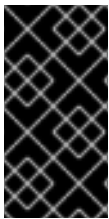
- **Installing a cluster on OpenStack on your own infrastructure** You can install OpenShift

Container Platform on user-provisioned RHOSP infrastructure. By using this installation method, you can integrate your cluster with existing infrastructure and modifications. For installations on user-provisioned infrastructure, you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. You can use the provided Ansible playbooks to assist with the deployment process.

- [Installing a cluster on OpenStack with Kuryr on your own infrastructure](#) You can install OpenShift Container Platform on user-provisioned RHOSP infrastructure that uses Kuryr SDN.

21.1.3. Scanning RHOSP endpoints for legacy HTTPS certificates

Beginning with OpenShift Container Platform 4.10, HTTPS certificates must contain subject alternative name (SAN) fields. Run the following script to scan each HTTPS endpoint in a Red Hat OpenStack Platform (RHOSP) catalog for legacy certificates that only contain the **CommonName** field.



IMPORTANT

OpenShift Container Platform does not check the underlying RHOSP infrastructure for legacy certificates prior to installation or updates. Use the provided script to check for these certificates yourself. Failing to update legacy certificates prior to installing or updating a cluster will result in cluster dysfunction.

Prerequisites

- On the machine where you run the script, have the following software:
 - Bash version 4.0 or greater
 - **grep**
 - [OpenStack client](#)
 - **jq**
 - [OpenSSL version 1.1.1l or greater](#)
- Populate the machine with RHOSP credentials for the target cloud.

Procedure

1. Save the following script to your machine:

```
#!/usr/bin/env bash

set -Eeuo pipefail

declare catalog san
catalog="$(mktemp)"
san="$(mktemp)"
readonly catalog san

declare invalid=0

openstack catalog list --format json --column Name --column Endpoints \
| jq -r '[] | .Name as $name | .Endpoints[] | select(.interface=="public") | [$name, .interface,
```

```

.url] | join(" ") \
| sort \
> "$catalog"

while read -r name interface url; do
# Ignore HTTP
if [[ ${url#"http://"} != "$url" ]]; then
continue
fi

# Remove the schema from the URL
noschema=${url#"https://"}

# If the schema was not HTTPS, error
if [[ "$noschema" == "$url" ]]; then
echo "ERROR (unknown schema): $name $interface $url"
exit 2
fi

# Remove the path and only keep host and port
noschema=${noschema%/*}
host=${noschema%:*}
port=${noschema##*:*}

# Add the port if was implicit
if [[ "$port" == "$host" ]]; then
port=443
fi

# Get the SAN fields
openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
| openssl x509 -noout -ext subjectAltName \
> "$san"

# openssl returns the empty string if no SAN is found.
# If a SAN is found, openssl is expected to return something like:
#
# X509v3 Subject Alternative Name:
#   DNS:standalone, DNS:osp1, IP Address:192.168.2.1, IP Address:10.254.1.2
if [[ "$(grep -c "Subject Alternative Name" "$san" || true)" -gt 0 ]]; then
echo "PASS: $name $interface $url"
else
invalid=$((invalid+1))
echo "INVALID: $name $interface $url"
fi
done < "$catalog"

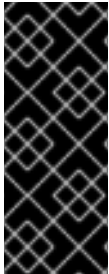
# clean up temporary files
rm "$catalog" "$san"

if [[ $invalid -gt 0 ]]; then
echo "${invalid} legacy certificates were detected. Update your certificates to include a SAN
field."
exit 1

```

```
else
  echo "All HTTPS certificates for this cloud are valid."
fi
```

2. Run the script.
3. Replace any certificates that the script reports as **INVALID** with certificates that contain SAN fields.



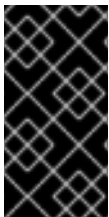
IMPORTANT

You must replace all legacy HTTPS certificates before you install OpenShift Container Platform 4.10 or update a cluster to that version. Legacy certificates will be rejected with the following message:

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

21.1.3.1. Scanning RHOSP endpoints for legacy HTTPS certificates manually

Beginning with OpenShift Container Platform 4.10, HTTPS certificates must contain subject alternative name (SAN) fields. If you do not have access to the prerequisite tools that are listed in "Scanning RHOSP endpoints for legacy HTTPS certificates", perform the following steps to scan each HTTPS endpoint in a Red Hat OpenStack Platform (RHOSP) catalog for legacy certificates that only contain the **CommonName** field.



IMPORTANT

OpenShift Container Platform does not check the underlying RHOSP infrastructure for legacy certificates prior to installation or updates. Use the following steps to check for these certificates yourself. Failing to update legacy certificates prior to installing or updating a cluster will result in cluster dysfunction.

Procedure

1. On a command line, run the following command to view the URL of RHOSP public endpoints:

```
$ openstack catalog list
```

Record the URL for each HTTPS endpoint that the command returns.

2. For each public endpoint, note the host and the port.

TIP

Determine the host of an endpoint by removing the scheme, the port, and the path.

3. For each endpoint, run the following commands to extract the SAN field of the certificate:

- a. Set a **host** variable:

```
$ host=<host_name>
```

- b. Set a **port** variable:

```
$ port=<port_number>
```

If the URL of the endpoint does not have a port, use the value **443**.

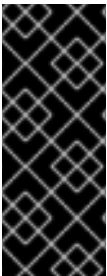
- c. Retrieve the SAN field of the certificate:

```
$ openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
| openssl x509 -noout -ext subjectAltName
```

Example output

```
X509v3 Subject Alternative Name:
DNS:your.host.example.net
```

For each endpoint, look for output that resembles the previous example. If there is no output for an endpoint, the certificate of that endpoint is invalid and must be re-issued.



IMPORTANT

You must replace all legacy HTTPS certificates before you install OpenShift Container Platform 4.10 or update a cluster to that version. Legacy certificates are rejected with the following message:

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

21.2. PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV OR OVS-DPDK ON OPENSTACK

Before you install a OpenShift Container Platform cluster that uses single-root I/O virtualization (SR-IOV) or Open vSwitch with the Data Plane Development Kit (OVS-DPDK) on Red Hat OpenStack Platform (RHOSP), you must understand the requirements for each technology and then perform preparatory tasks.

21.2.1. Requirements for clusters on RHOSP that use either SR-IOV or OVS-DPDK

If you use SR-IOV or OVS-DPDK with your deployment, you must meet the following requirements:

- RHOSP compute nodes must use a flavor that supports huge pages.

21.2.1.1. Requirements for clusters on RHOSP that use SR-IOV

To use single-root I/O virtualization (SR-IOV) with your deployment, you must meet the following requirements:

- [Plan your Red Hat OpenStack Platform \(RHOSP\) SR-IOV deployment](#) .
- OpenShift Container Platform must support the NICs that you use. For a list of supported NICs, see "About Single Root I/O Virtualization (SR-IOV) hardware networks" in the "Hardware networks" subsection of the "Networking" documentation.
- For each node that will have an attached SR-IOV NIC, your RHOSP cluster must have:

- One instance from the RHOSP quota
- One port attached to the machines subnet
- One port for each SR-IOV Virtual Function
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space
- SR-IOV deployments often employ performance optimizations, such as dedicated or isolated CPUs. For maximum performance, configure your underlying RHOSP deployment to use these optimizations, and then run OpenShift Container Platform compute machines on the optimized infrastructure.
 - For more information about configuring performant RHOSP compute nodes, see [Configuring Compute nodes for performance](#).

21.2.1.2. Requirements for clusters on RHOSP that use OVS-DPDK

To use Open vSwitch with the Data Plane Development Kit (OVS-DPDK) with your deployment, you must meet the following requirements:

- Plan your Red Hat OpenStack Platform (RHOSP) OVS-DPDK deployment by referring to [Planning your OVS-DPDK deployment](#) in the Network Functions Virtualization Planning and Configuration Guide.
- Configure your RHOSP OVS-DPDK deployment according to [Configuring an OVS-DPDK deployment](#) in the Network Functions Virtualization Planning and Configuration Guide.

21.2.2. Preparing to install a cluster that uses SR-IOV

You must configure RHOSP before you install a cluster that uses SR-IOV on it.

21.2.2.1. Creating SR-IOV networks for compute machines

If your Red Hat OpenStack Platform (RHOSP) deployment supports [single root I/O virtualization \(SR-IOV\)](#), you can provision SR-IOV networks that compute machines run on.



NOTE

The following instructions entail creating an external flat network and an external, VLAN-based network that can be attached to a compute machine. Depending on your RHOSP deployment, other network types might be required.

Prerequisites

- Your cluster supports SR-IOV.



NOTE

If you are unsure about what your cluster supports, review the OpenShift Container Platform SR-IOV hardware networks documentation.

- You created radio and uplink provider networks as part of your RHOSP deployment. The names **radio** and **uplink** are used in all example commands to represent these networks.

Procedure

1. On a command line, create a radio RHOSP network:

```
$ openstack network create radio --provider-physical-network radio --provider-network-type flat --external
```

2. Create an uplink RHOSP network:

```
$ openstack network create uplink --provider-physical-network uplink --provider-network-type vlan --external
```

3. Create a subnet for the radio network:

```
$ openstack subnet create --network radio --subnet-range <radio_network_subnet_range> radio
```

4. Create a subnet for the uplink network:

```
$ openstack subnet create --network uplink --subnet-range <uplink_network_subnet_range> uplink
```

21.2.3. Preparing to install a cluster that uses OVS-DPDK

You must configure RHOSP before you install a cluster that uses SR-IOV on it.

- Complete [Creating a flavor and deploying an instance for OVS-DPDK](#) before you install a cluster on RHOSP.

After you perform preinstallation tasks, install your cluster by following the most relevant OpenShift Container Platform on RHOSP installation instructions. Then, perform the tasks under "Next steps" on this page.

21.2.4. Next steps

- For either type of deployment:
 - [Configure the Node Tuning Operator with huge pages support](#) .
- To complete SR-IOV configuration after you deploy your cluster:
 - [Install the SR-IOV Operator](#) .
 - [Configure your SR-IOV network device](#) .
 - [Create SR-IOV compute machines](#) .
- Consult the following references after you deploy your cluster to improve its performance:
 - [A test pod template for clusters that use OVS-DPDK on OpenStack](#) .
 - [A test pod template for clusters that use SR-IOV on OpenStack](#) .
 - [A performance profile template for clusters that use OVS-DPDK on OpenStack](#) .

21.3. INSTALLING A CLUSTER ON OPENSTACK WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a customized cluster on Red Hat OpenStack Platform (RHOSP). To customize the installation, modify parameters in the **install-config.yaml** before you install the cluster.

21.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update processes](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You verified that OpenShift Container Platform 4.12 is compatible with your RHOSP version by using the [Supported platforms for OpenShift clusters](#) section. You can also compare platform support across different versions by viewing the [OpenShift Container Platform on RHOSP support matrix](#).
- You have a storage service installed in RHOSP, such as block storage (Cinder) or object storage (Swift). Object storage is the recommended storage technology for OpenShift Container Platform registry cluster deployment. For more information, see [Optimizing storage](#).
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see [Recommended practices for scaling the cluster](#).
- You have the metadata service enabled in RHOSP.

21.3.2. Resource guidelines for installing OpenShift Container Platform on RHOSP

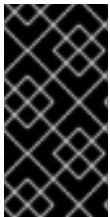
To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 21.1. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

Resource	Value
Floating IP addresses	3
Ports	15
Routers	1
Subnets	1
RAM	88 GB
vCPUs	22
Volume storage	275 GB

Resource	Value
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

21.3.2.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.3.2.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota

- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

21.3.2.3. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

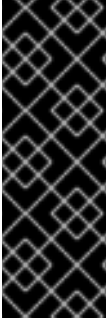


IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

21.3.4. Enabling Swift on RHOSP

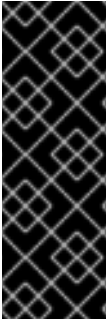
Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.



IMPORTANT

If the [Red Hat OpenStack Platform \(RHOSP\) object storage service](#), commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.



IMPORTANT

RHOSP 17 sets the **rgw_max_attr_size** parameter of Ceph RGW to 256 characters. This setting causes issues with uploading container images to the OpenShift Container Platform registry. You must set the value of **rgw_max_attr_size** to at least 1024 characters.

Before installation, check if your RHOSP deployment is affected by this problem. If it is, reconfigure Ceph RGW.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On [Ceph RGW](#), the **account in url** option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

```
$ openstack role add --user <user> --project <project> swiftoperator
```

Your RHOSP deployment can now use Swift for the image registry.

21.3.5. Configuring an image registry with custom storage on clusters that run on RHOSP

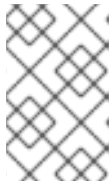
After you install a cluster on Red Hat OpenStack Platform (RHOSP), you can use a Cinder volume that is in a specific availability zone for registry storage.

Procedure

1. Create a YAML file that specifies the storage class and availability zone to use. For example:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: custom-csi-storageclass
provisioner: cinder.csi.openstack.org
volumeBindingMode: WaitForFirstConsumer
```

```
allowVolumeExpansion: true
parameters:
  availability: <availability_zone_name>
```



NOTE

OpenShift Container Platform does not verify the existence of the availability zone you choose. Verify the name of the availability zone before you apply the configuration.

- From a command line, apply the configuration:

```
$ oc apply -f <storage_class_file_name>
```

Example output

```
storageclass.storage.k8s.io/custom-csi-storageclass created
```

- Create a YAML file that specifies a persistent volume claim (PVC) that uses your storage class and the **openshift-image-registry** namespace. For example:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc-imageregistry
  namespace: openshift-image-registry 1
  annotations:
    imageregistry.openshift.io: "true"
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi 2
  storageClassName: <your_custom_storage_class> 3
```

- 1** Enter the namespace **openshift-image-registry**. This namespace allows the Cluster Image Registry Operator to consume the PVC.
- 2** Optional: Adjust the volume size.
- 3** Enter the name of the storage class that you created.

- From a command line, apply the configuration:

```
$ oc apply -f <pvc_file_name>
```

Example output

```
persistentvolumeclaim/csi-pvc-imageregistry created
```

- Replace the original persistent volume claim in the image registry configuration with the new claim:

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type 'json' -p='[{"op":
"replace", "path": "/spec/storage/pvc/claim", "value": "csi-pvc-imageregistry"}]'
```

Example output

```
config.imageregistry.operator.openshift.io/cluster patched
```

Over the next several minutes, the configuration is updated.

Verification

To confirm that the registry is using the resources that you defined:

- Verify that the PVC claim value is identical to the name that you provided in your PVC definition:

```
$ oc get configs.imageregistry.operator.openshift.io/cluster -o yaml
```

Example output

```
...
status:
  ...
  managementState: Managed
  pvc:
    claim: csi-pvc-imageregistry
  ...
```

- Verify that the status of the PVC is **Bound**:

```
$ oc get pvc -n openshift-image-registry csi-pvc-imageregistry
```

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
csi-pvc-imageregistry	Bound	pvc-72a8f9c9-f462-11e8-b6b6-fa163e18b7b5	100Gi	
RWO	custom-csi-storageclass	11m		

21.3.6. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- [Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries](#)

Procedure

- Using the RHOSP CLI, verify the name and ID of the 'External' network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+-----+-----+-----+
| ID              | Name      | Router Type |
+-----+-----+-----+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External   |
+-----+-----+-----+
```

A network with an external router type appears in the network list. If at least one does not, see [Creating a default floating IP network](#) and [Creating a default provider network](#).

IMPORTANT

If the external network's CIDR range overlaps one of the default network ranges, you must change the matching network ranges in the **install-config.yaml** file before you start the installation process.

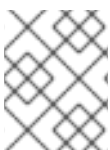
The default network ranges are:

Network	Range
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14



WARNING

If the installation program finds multiple networks with the same name, it sets one of them at random. To avoid this behavior, create unique names for resources in RHOSP.



NOTE

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see [Neutron trunk port](#).

21.3.7. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:

- If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.



IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in [a separate file](#) from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see [Config files](#) in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:

- Copy the certificate authority file to your machine.
- Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
 - a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
 - b. The current directory
 - c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**
 - d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**

The installation program searches for **clouds.yaml** in that order.

21.3.8. Setting OpenStack Cloud Controller Manager options

Optionally, you can edit the OpenStack Cloud Controller Manager (CCM) configuration for your cluster. This configuration controls how OpenShift Container Platform interacts with Red Hat OpenStack Platform (RHOSP).

For a complete list of configuration parameters, see the "OpenStack Cloud Controller Manager reference guide" page in the "Installing on OpenStack" documentation.

Procedure

1. If you have not already generated manifest files for your cluster, generate them by running the following command:

```
$ openshift-install --dir <destination_directory> create manifests
```

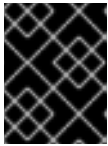
2. In a text editor, open the cloud-provider configuration manifest file. For example:

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. Modify the options according to the CCM reference guide. Configuring Octavia for load balancing is a common case for clusters that do not use Kuryr. For example:

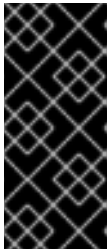
```
#...
[LoadBalancer]
use-octavia=true 1
lb-provider = "amphora" 2
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 3
create-monitor = True 4
monitor-delay = 10s 5
monitor-timeout = 10s 6
monitor-max-retries = 1 7
#...
```

- 1 This property enables Octavia integration.
- 2 This property sets the Octavia provider that your load balancer uses. It accepts **"ovn"** or **"amphora"** as values. If you choose to use OVN, you must also set **lb-method** to **SOURCE_IP_PORT**.
- 3 This property is required if you want to use multiple external networks with your cluster. The cloud provider creates floating IP addresses on the network that is specified here.
- 4 This property controls whether the cloud provider creates health monitors for Octavia load balancers. Set the value to **True** to create health monitors. As of RHOSP 16.2, this feature is only available for the Amphora provider.
- 5 This property sets the frequency with which endpoints are monitored. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
- 6 This property sets the time that monitoring requests are open before timing out. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
- 7 This property defines how many successful monitoring requests are required before a load balancer is marked as online. The value must be an integer. This property is required if the value of the **create-monitor** property is **True**.



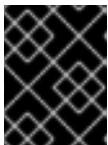
IMPORTANT

Prior to saving your changes, verify that the file is structured correctly. Clusters might fail if properties are not placed in the appropriate section.



IMPORTANT

You must set the value of the **create-monitor** property to **True** if you use services that have the value of the **.spec.externalTrafficPolicy** property set to **Local**. The OVN Octavia provider in RHOSP 16.2 does not support health monitors. Therefore, services that have **ETP** parameter values set to **Local** might not respond when the **lb-provider** value is set to **"ovn"**.



IMPORTANT

For installations that use Kuryr, Kuryr handles relevant services. There is no need to configure Octavia load balancing in the cloud provider.

4. Save the changes to the file and proceed with installation.

TIP

You can update your cloud provider configuration after you run the installer. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

After you save your changes, your cluster will take some time to reconfigure itself. The process is complete if none of your nodes have a **SchedulingDisabled** status.

21.3.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

21.3.10. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **openstack** as the platform to target.
- iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.
- iv. Specify the floating IP address to use for external access to the OpenShift API.
- v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

- vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.
 - vii. Enter a name for your cluster. The name must be 14 or fewer characters long.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

See [Installation configuration parameters section](#) for more information about the available parameters.

21.3.10.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

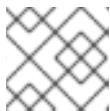
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
```

```

httpProxy: http://<username>:<pswd>@<ip>:<port> 1
httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

21.3.11. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

21.3.11.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 21.2. Required parameters

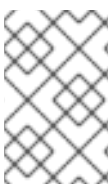
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev . The string must be 14 characters or fewer long.
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

21.3.11.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 21.3. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

21.3.11.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 21.4. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}

Parameter	Description	Values
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings

Parameter	Description	Values
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...

21.3.11.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 21.5. Additional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.rootVolume.size	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
compute.platform.openstack.rootVolume.type	For compute machines, the root volume's type.	String, for example performance .

Parameter	Description	Values
controlPlane.platform.openstack.rootVolume.size	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
controlPlane.platform.openstack.rootVolume.type	For control plane machines, the root volume's type.	String, for example performance .
platform.openstack.cloud	The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.	String, for example MyCloud .
platform.openstack.externalNetwork	The RHOSP external network name to be used for installation.	String, for example external .
platform.openstack.computeFlavor	The RHOSP flavor to use for control plane and compute machines. This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the type key in the platform.openstack.defaultMachinePlatform property. You can also set a flavor value for each machine pool individually.	String, for example m1.xlarge .

21.3.11.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 21.6. Optional RHOSP parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
compute.platform.openstack.additionalNetworkIDs	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .
compute.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .
compute.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
compute.platform.openstack.rootVolume.zones	For compute machines, the availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .

Parameter	Description	Values
compute.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	<p>A server group policy to apply to the machine pool. For example, soft-affinity.</p>
controlPlane.platform.openstack.additionalNetworkIDs	<p>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</p> <p>Additional networks that are attached to a control plane machine are also attached to the bootstrap node.</p>	<p>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</p>
controlPlane.platform.openstack.additionalSecurityGroupIDs	<p>Additional security groups that are associated with control plane machines.</p>	<p>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</p>

Parameter	Description	Values
controlPlane.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
controlPlane.platform.openstack.rootVolume.zones	For control plane machines, the availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .
controlPlane.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations, and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .

Parameter	Description	Values
platform.openstack.clusterOSImage	<p>The location from which the installation program downloads the RHCOS image.</p> <p>You must set this parameter to perform an installation in a restricted network.</p>	<p>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.</p> <p>For example, http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d. The value can also be the name of an existing Glance image, for example my-rhcos.</p>
platform.openstack.clusterOSImageProperties	<p>Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if platform.openstack.clusterOSImage is set to an existing Glance image.</p> <p>You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the hw_scsi_model property value to virtio-scsi and the hw_disk_bus value to scsi.</p> <p>You can also use this property to enable the QEMU guest agent by including the hw_qemu_guest_agent property with a value of yes.</p>	<p>A list of key-value string pairs. For example, [{"hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"}].</p>
platform.openstack.defaultMachinePlatform	<p>The default machine pool platform configuration.</p>	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</p>	<p>An IP address, for example 128.0.0.1.</p>

Parameter	Description	Values
platform.openstack.apiFloatingIP	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.	An IP address, for example 128.0.0.1 .
platform.openstack.externalDNS	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, ["8.8.8.8", "192.168.1.12"] .
platform.openstack.machinesSubnet	<p>The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</p> <p>The first item in networking.machineNetwork must match the value of machinesSubnet.</p> <p>If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</p>	A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

21.3.11.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.
- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.



NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.



IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

21.3.11.7. Deploying a cluster with bare metal machines

If you want your cluster to use bare metal machines, modify the **install-config.yaml** file. Your cluster can have both control plane and compute machines running on bare metal, or just compute machines.

Bare-metal compute machines are not supported on clusters that use Kuryr.



NOTE

Be sure that your **install-config.yaml** file reflects whether the RHOSP network that you use for bare metal workers supports floating IP addresses or not.

Prerequisites

- The RHOSP [Bare Metal service \(Ironic\)](#) is enabled and accessible via the RHOSP Compute API.
- Bare metal is available as [a RHOSP flavor](#).
- If your cluster runs on an RHOSP version that is more than 16.1.6 and less than 16.2.4, bare metal workers do not function due to a [known issue](#) that causes the metadata service to be unavailable for services on OpenShift Container Platform nodes.
- The RHOSP network supports both VM and bare metal server attachment.
- Your network configuration does not rely on a provider network. Provider networks are not supported.
- If you want to deploy the machines on a pre-existing network, a RHOSP subnet is provisioned.
- If you want to deploy the machines on an installer-provisioned network, the RHOSP Bare Metal service (Ironic) is able to listen for and interact with Preboot eXecution Environment (PXE) boot machines that run on tenant networks.

- You created an **install-config.yaml** file as part of the OpenShift Container Platform installation process.

Procedure

1. In the **install-config.yaml** file, edit the flavors for machines:
 - a. If you want to use bare-metal control plane machines, change the value of **controlPlane.platform.openstack.type** to a bare metal flavor.
 - b. Change the value of **compute.platform.openstack.type** to a bare metal flavor.
 - c. If you want to deploy your machines on a pre-existing network, change the value of **platform.openstack.machinesSubnet** to the RHOSP subnet UUID of the network. Control plane and compute machines must use the same subnet.

An example bare metal install-config.yaml file

```
controlPlane:
  platform:
    openstack:
      type: <bare_metal_control_plane_flavor> 1
  ...

compute:
  - architecture: amd64
    hyperthreading: Enabled
    name: worker
    platform:
      openstack:
        type: <bare_metal_compute_flavor> 2
      replicas: 3
  ...

platform:
  openstack:
    machinesSubnet: <subnet_UUID> 3
  ...
```

- 1 If you want to have bare-metal control plane machines, change this value to a bare metal flavor.
- 2 Change this value to a bare metal flavor to use for compute machines.
- 3 If you want to use a pre-existing network, change this value to the UUID of the RHOSP subnet.

Use the updated **install-config.yaml** file to complete the installation process. The compute machines that are created during deployment use the flavor that you added to the file.



NOTE

The installer may time out while waiting for bare metal machines to boot.

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

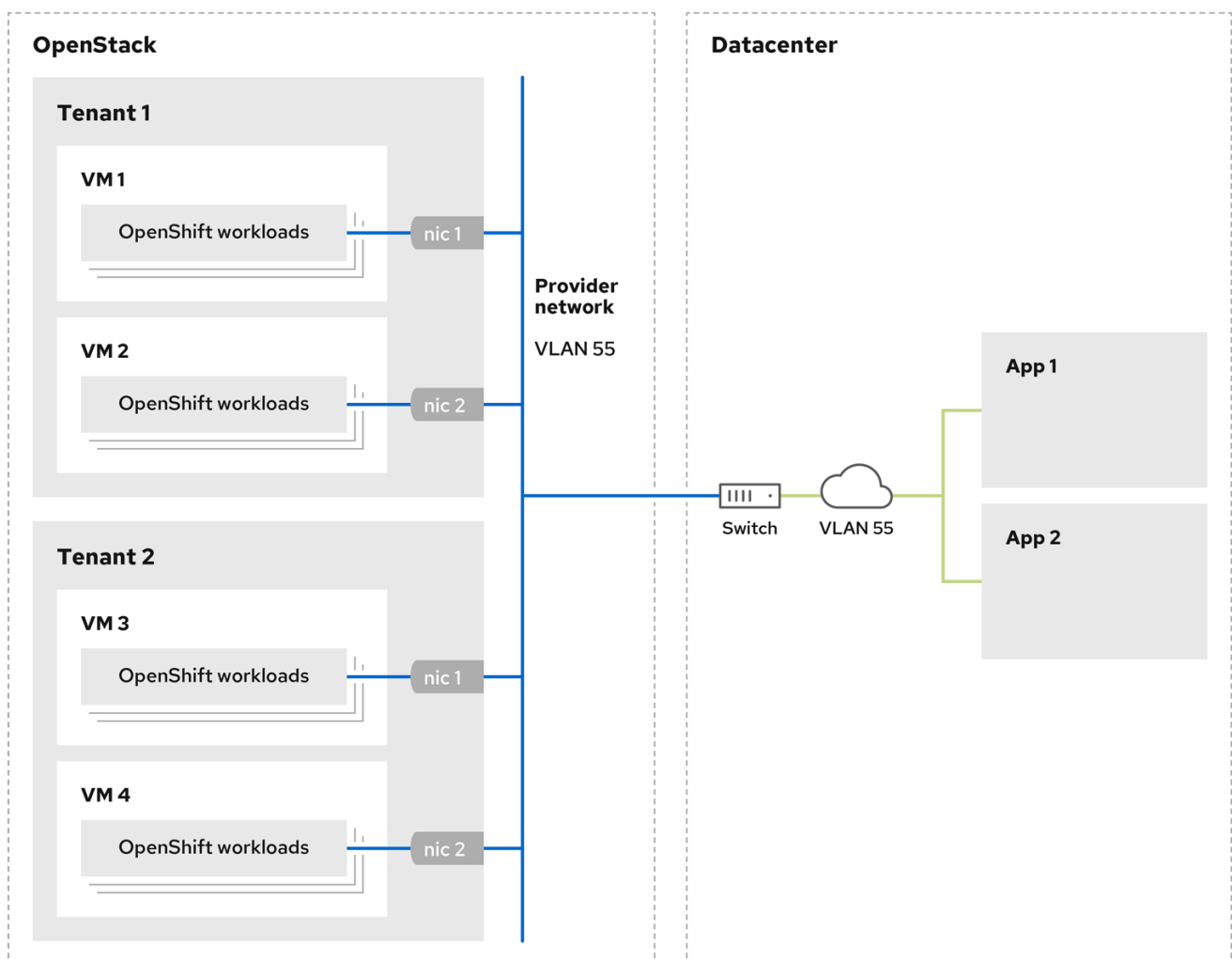
```
$ ./openshift-install wait-for install-complete --log-level debug
```

21.3.11.8. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



I70_OpenShift_0621

OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).



NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in [the RHOSP documentation](#).

21.3.11.8.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The [RHOSP networking service \(Neutron\) is enabled](#) and accessible through the RHOSP networking API.
- The RHOSP networking service has the [port security and allowed address pairs extensions enabled](#).
- The provider network can be shared with other tenants.

TIP

Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

TIP

To create a network for a project that is named "openshift," enter the following command

```
$ openstack network create --project openshift
```

To create a subnet for a project that is named "openshift," enter the following command

```
$ openstack subnet create --project openshift
```

To learn more about creating networks on RHOSP, read [the provider networks documentation](#).

If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.



IMPORTANT

Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.
Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

```
$ openstack subnet create --dhcp --host-route
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- Optional: To secure the network, create [role-based access control \(RBAC\)](#) rules that limit network access to a single project.

21.3.11.8.2. Deploying a cluster that has a primary interface on a provider network

You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.



IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: 1
```

```

- 192.0.2.13
ingressVIPs: 2
- 192.0.2.23
machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
# ...
networking:
  machineNetwork:
    - cidr: 192.0.2.0/24

```

- 1 2** In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.



WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

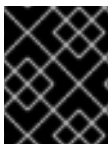
TIP

You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see [Understanding multiple networks](#).

21.3.11.9. Sample customized install-config.yaml file for RHOSP

This sample **install-config.yaml** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.



IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

```

apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
  - name: worker

```

```

platform:
  openstack:
    type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
  fips: false
  pullSecret: '{"auths": ...}'
  sshKey: ssh-ed25519 AAAA...

```

21.3.12. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

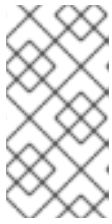
Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

21.3.13. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

21.3.13.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```



NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your `/etc/hosts` file:

- `<api_floating_ip> api.<cluster_name>.<base_domain>`
- `<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>`
- `application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>`

The cluster domain names in the `/etc/hosts` file grant access to the web console and the monitoring interface of your cluster locally. You can also use the `kubectl` or `oc`. You can access the user applications by using the additional entries pointing to the `<application_floating_ip>`. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

4. Add the FIPs to the `install-config.yaml` file as the values of the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.apiFloatingIP`

If you use these values, you must also enter an external network as the value of the `platform.openstack.externalNetwork` parameter in the `install-config.yaml` file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

21.3.13.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `install-config.yaml` file, do not define the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.apiFloatingIP`

If you cannot provide an external network, you can also leave `platform.openstack.externalNetwork` blank. If you do not provide a value for `platform.openstack.externalNetwork`, a router is not created for

you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

21.3.14. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

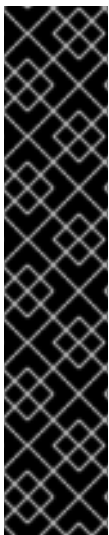
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

21.3.15. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

- View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

- View your cluster's version:

```
$ oc get clusterversion
```

- View your Operators' status:

```
$ oc get clusteroperator
```

- View all running pods in the cluster:

```
$ oc get pods -A
```

21.3.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

- Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

system:admin

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

21.3.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

21.3.18. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- If you need to enable external access to node ports, [configure ingress cluster traffic by using a node port](#).
- If you did not configure RHOSP to accept application traffic over floating IP addresses, [configure RHOSP access with floating IP addresses](#).

21.4. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR



IMPORTANT

Kuryr is a deprecated feature. Deprecated functionality is still included in OpenShift Container Platform and continues to be supported; however, it will be removed in a future release of this product and is not recommended for new deployments.

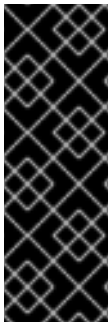
For the most recent list of major functionality that has been deprecated or removed within OpenShift Container Platform, refer to the *Deprecated and removed features* section of the OpenShift Container Platform release notes.

In OpenShift Container Platform version 4.12, you can install a customized cluster on Red Hat OpenStack Platform (RHOSP) that uses Kuryr SDN. To customize the installation, modify parameters in the `install-config.yaml` before you install the cluster.

21.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You verified that OpenShift Container Platform 4.12 is compatible with your RHOSP version by using the [Supported platforms for OpenShift clusters](#) section. You can also compare platform support across different versions by viewing the [OpenShift Container Platform on RHOSP support matrix](#).
- You have a storage service installed in RHOSP, such as block storage (Cinder) or object storage (Swift). Object storage is the recommended storage technology for OpenShift Container Platform registry cluster deployment. For more information, see [Optimizing storage](#).
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see [Recommended practices for scaling the cluster](#).

21.4.2. About Kuryr SDN



IMPORTANT

Kuryr is a deprecated feature. Deprecated functionality is still included in OpenShift Container Platform and continues to be supported; however, it will be removed in a future release of this product and is not recommended for new deployments.

For the most recent list of major functionality that has been deprecated or removed within OpenShift Container Platform, refer to the *Deprecated and removed features* section of the OpenShift Container Platform release notes.

[Kuryr](#) is a container network interface (CNI) plugin solution that uses the [Neutron](#) and [Octavia](#) Red Hat OpenStack Platform (RHOSP) services to provide networking for pods and Services.

Kuryr and OpenShift Container Platform integration is primarily designed for OpenShift Container Platform clusters running on RHOSP VMs. Kuryr improves the network performance by plugging OpenShift Container Platform pods into RHOSP SDN. In addition, it provides interconnectivity between pods and RHOSP virtual instances.

Kuryr components are installed as pods in OpenShift Container Platform using the **openshift-kuryr** namespace:

- **kuryr-controller** - a single service instance installed on a **master** node. This is modeled in OpenShift Container Platform as a **Deployment** object.
- **kuryr-cni** - a container installing and configuring Kuryr as a CNI driver on each OpenShift Container Platform node. This is modeled in OpenShift Container Platform as a **DaemonSet** object.

The Kuryr controller watches the OpenShift Container Platform API server for pod, service, and namespace create, update, and delete events. It maps the OpenShift Container Platform API calls to corresponding objects in Neutron and Octavia. This means that every network solution that implements the Neutron trunk port functionality can be used to back OpenShift Container Platform via Kuryr. This includes open source solutions such as Open vSwitch (OVS) and Open Virtual Network (OVN) as well as Neutron-compatible commercial SDNs.

Kuryr is recommended for OpenShift Container Platform deployments on encapsulated RHOSP tenant networks to avoid double encapsulation, such as running an encapsulated OpenShift Container Platform SDN over an RHOSP network.

If you use provider networks or tenant VLANs, you do not need to use Kuryr to avoid double encapsulation. The performance benefit is negligible. Depending on your configuration, though, using Kuryr to avoid having two overlays might still be beneficial.

Kuryr is not recommended in deployments where all of the following criteria are true:

- The RHOSP version is less than 16.
- The deployment uses UDP services, or a large number of TCP services on few hypervisors.

or

- The **ovn-octavia** Octavia driver is disabled.
- The deployment uses a large number of TCP services on few hypervisors.

21.4.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr

When using Kuryr SDN, the pods, services, namespaces, and network policies are using resources from the RHOSP quota; this increases the minimum requirements. Kuryr also has some additional requirements on top of what a default install requires.

Use the following quota to satisfy a default cluster's minimum requirements:

Table 21.7. Recommended resources for a default OpenShift Container Platform cluster on RHOSP with Kuryr

Resource	Value
Floating IP addresses	3 - plus the expected number of Services of LoadBalancer type
Ports	1500 - 1 needed per Pod
Routers	1
Subnets	250 - 1 needed per Namespace/Project
Networks	250 - 1 needed per Namespace/Project
RAM	112 GB
vCPUs	28
Volume storage	275 GB
Instances	7

Resource	Value
Security groups	250 - 1 needed per Service and per NetworkPolicy
Security group rules	1000
Server groups	2 - plus 1 for each additional availability zone in each machine pool
Load balancers	100 - 1 needed per Service
Load balancer listeners	500 - 1 needed per Service-exposed port
Load balancer pools	500 - 1 needed per Service-exposed port

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



IMPORTANT

If you are using Red Hat OpenStack Platform (RHOSP) version 16 with the Amphora driver rather than the OVN Octavia driver, security groups are associated with service accounts instead of user projects.

Take the following notes into consideration when setting resources:

- The number of ports that are required is larger than the number of pods. Kuryr uses ports pools to have pre-created ports ready to be used by pods and speed up the pods' booting time.
- Each network policy is mapped into an RHOSP security group, and depending on the **NetworkPolicy** spec, one or more rules are added to the security group.
- Each service is mapped to an RHOSP load balancer. Consider this requirement when estimating the number of security groups required for the quota.
If you are using RHOSP version 15 or earlier, or the **ovn-octavia driver**, each load balancer has a security group with the user project.
- The quota does not account for load balancer resources (such as VM resources), but you must consider these resources when you decide the RHOSP deployment's size. The default installation will have more than 50 load balancers; the clusters must be able to accommodate them.
If you are using RHOSP version 16 with the OVN Octavia driver enabled, only one load balancer VM is generated; services are load balanced through OVN flows.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

To enable Kuryr SDN, your environment must meet the following requirements:

- Run RHOSP 13+.
- Have Overcloud with Octavia.
- Use Neutron Trunk ports extension.
- Use **openvswitch** firewall driver if ML2/OVS Neutron driver is used instead of **ovs-hybrid**.

21.4.3.1. Increasing quota

When using Kuryr SDN, you must increase quotas to satisfy the Red Hat OpenStack Platform (RHOSP) resources used by pods, services, namespaces, and network policies.

Procedure

- Increase the quotas for a project by running the following command:

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets 250 --networks 250 <project>
```

21.4.3.2. Configuring Neutron

Kuryr CNI leverages the Neutron Trunks extension to plug containers into the Red Hat OpenStack Platform (RHOSP) SDN, so you must use the **trunks** extension for Kuryr to properly work.

In addition, if you leverage the default ML2/OVS Neutron driver, the firewall must be set to **openvswitch** instead of **ovs_hybrid** so that security groups are enforced on trunk subports and Kuryr can properly handle network policies.

21.4.3.3. Configuring Octavia

Kuryr SDN uses Red Hat OpenStack Platform (RHOSP)'s Octavia LBaaS to implement OpenShift Container Platform services. Thus, you must install and configure Octavia components in RHOSP to use Kuryr SDN.

To enable Octavia, you must include the Octavia service during the installation of the RHOSP Overcloud, or upgrade the Octavia service if the Overcloud already exists. The following steps for enabling Octavia apply to both a clean install of the Overcloud or an Overcloud update.



NOTE

The following steps only capture the key pieces required during the [deployment of RHOSP](#) when dealing with Octavia. It is also important to note that [registry methods](#) vary.

This example uses the local registry method.

Procedure

1. If you are using the local registry, create a template to upload the images to the registry. For example:


```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
--tag-from-label {version}-{product-version} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

2. Verify that the **local_registry_images.yaml** file contains the Octavia images. For example:

```
...
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```



NOTE

The Octavia container versions vary depending upon the specific RHOSP release installed.

3. Pull the container images from **registry.redhat.io** to the Undercloud node:

```
(undercloud) $ sudo openstack overcloud container image upload \
--config-file /home/stack/local_registry_images.yaml \
--verbose
```

This may take some time depending on the speed of your network and Undercloud disk.

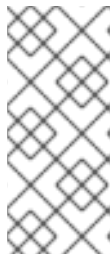
4. Install or update your Overcloud environment with Octavia:

```
$ openstack overcloud deploy --templates \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
-e octavia_timeouts.yaml
```



NOTE

This command only includes the files associated with Octavia; it varies based on your specific installation of RHOSP. See the RHOSP documentation for further information. For more information on customizing your Octavia installation, see [installation of Octavia using Director](#).

**NOTE**

When leveraging Kuryr SDN, the OpenShift installation requires the Neutron **trunk** extension. This is available by default on director deployments. Use the **openvswitch** firewall instead of the default **ovs-hybrid** when the Neutron backend is ML2/OVS. There is no need for modifications if the backend is ML2/OVN.

21.4.3.3.1. The Octavia OVN Driver

Octavia supports multiple provider drivers through the Octavia API.

To see all available Octavia provider drivers, on a command line, enter:

```
$ openstack loadbalancer provider list
```

Example output

```
+-----+-----+
| name | description |
+-----+-----+
| amphora | The Octavia Amphora driver. |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn | Octavia OVN driver. |
+-----+-----+
```

Beginning with RHOSP version 16, the Octavia OVN provider driver (**ovn**) is supported on OpenShift Container Platform on RHOSP deployments.

ovn is an integration driver for the load balancing that Octavia and OVN provide. It supports basic load balancing capabilities, and is based on OpenFlow rules. The driver is automatically enabled in Octavia by Director on deployments that use OVN Neutron ML2.

The Amphora provider driver is the default driver. If **ovn** is enabled, however, Kuryr uses it.

If Kuryr uses **ovn** instead of Amphora, it offers the following benefits:

- Decreased resource requirements. Kuryr does not require a load balancer VM for each service.
- Reduced network latency.
- Increased service creation speed by using OpenFlow rules instead of a VM for each service.
- Distributed load balancing actions across all nodes instead of centralized on Amphora VMs.

You can [configure your cluster to use the Octavia OVN driver](#) after your RHOSP cloud is upgraded from version 13 to version 16.

21.4.3.4. Known limitations of installing with Kuryr

Using OpenShift Container Platform with Kuryr SDN has several known limitations.

RHOSP general limitations

Using OpenShift Container Platform with Kuryr SDN has several limitations that apply to all versions and environments:

- **Service** objects with the **NodePort** type are not supported.
- Clusters that use the OVN Octavia provider driver support **Service** objects for which the **.spec.selector** property is unspecified only if the **.subsets.addresses** property of the **Endpoints** object includes the subnet of the nodes or pods.
- If the subnet on which machines are created is not connected to a router, or if the subnet is connected, but the router has no external gateway set, Kuryr cannot create floating IPs for **Service** objects with type **LoadBalancer**.
- Configuring the **sessionAffinity=ClientIP** property on **Service** objects does not have an effect. Kuryr does not support this setting.

RHOSP version limitations

Using OpenShift Container Platform with Kuryr SDN has several limitations that depend on the RHOSP version.

- RHOSP versions before 16 use the default Octavia load balancer driver (Amphora). This driver requires that one Amphora load balancer VM is deployed per OpenShift Container Platform service. Creating too many services can cause you to run out of resources. Deployments of later versions of RHOSP that have the OVN Octavia driver disabled also use the Amphora driver. They are subject to the same resource concerns as earlier versions of RHOSP.
- Kuryr SDN does not support automatic unidling by a service.

RHOSP upgrade limitations

As a result of the RHOSP upgrade process, the Octavia API might be changed, and upgrades to the Amphora images that are used for load balancers might be required.

You can address API changes on an individual basis.

If the Amphora image is upgraded, the RHOSP operator can handle existing load balancer VMs in two ways:

- Upgrade each VM by triggering a [load balancer failover](#).
- Leave responsibility for upgrading the VMs to users.

If the operator takes the first option, there might be short downtimes during failovers.

If the operator takes the second option, the existing load balancers will not support upgraded Octavia API features, like UDP listeners. In this case, users must recreate their Services to use these features.

21.4.3.5. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.4.3.6. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

21.4.3.7. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.4.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

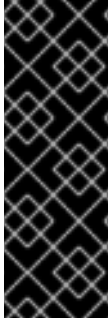


IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

21.4.5. Enabling Swift on RHOSP

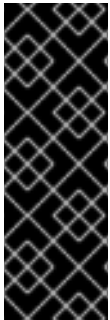
Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.



IMPORTANT

If [the Red Hat OpenStack Platform \(RHOSP\) object storage service](#), commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.



IMPORTANT

RHOSP 17 sets the **rgw_max_attr_size** parameter of Ceph RGW to 256 characters. This setting causes issues with uploading container images to the OpenShift Container Platform registry. You must set the value of **rgw_max_attr_size** to at least 1024 characters.

Before installation, check if your RHOSP deployment is affected by this problem. If it is, reconfigure Ceph RGW.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On [Ceph RGW](#), the **account in url** option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

```
$ openstack role add --user <user> --project <project> swiftoperator
```

Your RHOSP deployment can now use Swift for the image registry.

21.4.6. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- [Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries](#)

Procedure

- Using the RHOSP CLI, verify the name and ID of the 'External' network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+-----+-----+-----+
| ID                | Name          | Router Type |
+-----+-----+-----+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+-----+-----+-----+
```

A network with an external router type appears in the network list. If at least one does not, see [Creating a default floating IP network](#) and [Creating a default provider network](#).

IMPORTANT

If the external network's CIDR range overlaps one of the default network ranges, you must change the matching network ranges in the **install-config.yaml** file before you start the installation process.

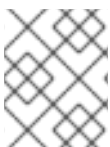
The default network ranges are:

Network	Range
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14



WARNING

If the installation program finds multiple networks with the same name, it sets one of them at random. To avoid this behavior, create unique names for resources in RHOSP.



NOTE

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see [Neutron trunk port](#).

21.4.7. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:

- If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.



IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in [a separate file](#) from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see [Config files](#) in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:

- Copy the certificate authority file to your machine.
- Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
 - a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
 - b. The current directory
 - c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**
 - d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**

The installation program searches for **clouds.yaml** in that order.

21.4.8. Setting OpenStack Cloud Controller Manager options

Optionally, you can edit the OpenStack Cloud Controller Manager (CCM) configuration for your cluster. This configuration controls how OpenShift Container Platform interacts with Red Hat OpenStack Platform (RHOSP).

For a complete list of configuration parameters, see the "OpenStack Cloud Controller Manager reference guide" page in the "Installing on OpenStack" documentation.

Procedure

1. If you have not already generated manifest files for your cluster, generate them by running the following command:

```
$ openshift-install --dir <destination_directory> create manifests
```

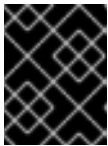
2. In a text editor, open the cloud-provider configuration manifest file. For example:

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. Modify the options according to the CCM reference guide. Configuring Octavia for load balancing is a common case for clusters that do not use Kuryr. For example:

```
#...
[LoadBalancer]
use-octavia=true 1
lb-provider = "amphora" 2
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 3
create-monitor = True 4
monitor-delay = 10s 5
monitor-timeout = 10s 6
monitor-max-retries = 1 7
#...
```


- 1 This property enables Octavia integration.
- 2 This property sets the Octavia provider that your load balancer uses. It accepts **"ovn"** or **"amphora"** as values. If you choose to use OVN, you must also set **lb-method** to **SOURCE_IP_PORT**.
- 3 This property is required if you want to use multiple external networks with your cluster. The cloud provider creates floating IP addresses on the network that is specified here.
- 4 This property controls whether the cloud provider creates health monitors for Octavia load balancers. Set the value to **True** to create health monitors. As of RHOSP 16.2, this feature is only available for the Amphora provider.
- 5 This property sets the frequency with which endpoints are monitored. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
- 6 This property sets the time that monitoring requests are open before timing out. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
- 7 This property defines how many successful monitoring requests are required before a load balancer is marked as online. The value must be an integer. This property is required if the value of the **create-monitor** property is **True**.



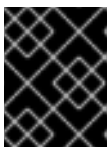
IMPORTANT

Prior to saving your changes, verify that the file is structured correctly. Clusters might fail if properties are not placed in the appropriate section.



IMPORTANT

You must set the value of the **create-monitor** property to **True** if you use services that have the value of the **.spec.externalTrafficPolicy** property set to **Local**. The OVN Octavia provider in RHOSP 16.2 does not support health monitors. Therefore, services that have **ETP** parameter values set to **Local** might not respond when the **lb-provider** value is set to **"ovn"**.



IMPORTANT

For installations that use Kuryr, Kuryr handles relevant services. There is no need to configure Octavia load balancing in the cloud provider.

4. Save the changes to the file and proceed with installation.

TIP

You can update your cloud provider configuration after you run the installer. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

After you save your changes, your cluster will take some time to reconfigure itself. The process is complete if none of your nodes have a **SchedulingDisabled** status.

21.4.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

21.4.10. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

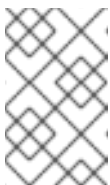
```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **openstack** as the platform to target.
- iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.
- iv. Specify the floating IP address to use for external access to the OpenShift API.
- v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

- vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.
 - vii. Enter a name for your cluster. The name must be 14 or fewer characters long.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

21.4.10.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.



NOTE

Kuryr installations default to HTTP proxies.

Prerequisites

- For Kuryr installations on restricted networks that use the **Proxy** object, the proxy must be able to reply to the router that the cluster uses. To add a static route for the proxy configuration, from a command line as the root user, enter:

```
$ ip route add <cluster_network_cidr> via <installer_subnet_gateway>
```

- The restricted subnet must have a gateway that is defined and available to be linked to the **Router** resource that Kuryr creates.
- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

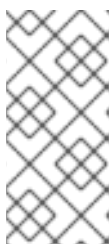
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

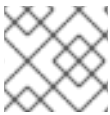


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

21.4.11. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

21.4.11.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 21.8. Required parameters

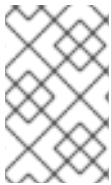
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .

Parameter	Description	Values
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev . The string must be 14 characters or fewer long.
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

21.4.11.2. Network configuration parameters


You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 21.9. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .


Parameter	Description	Values
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>


21.4.11.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:



Table 21.10. Optional parameters


Parameter	Description	Values
additionalTrustBundle	<p>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</p>	String
capabilities	<p>Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i>.</p>	String array

Parameter	Description	Values
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker

Parameter	Description	Values
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled

Parameter	Description	Values
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default	false or true

Parameter	Description	Values
	<p data-bbox="488 114 938 210">Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 264 593 1070" style="display: flex; align-items: center;">  <div data-bbox="671 271 938 1070" style="margin-left: 10px;"> <p data-bbox="671 271 863 300">IMPORTANT</p> <p data-bbox="671 338 938 1070">To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div data-bbox="488 1122 593 1317" style="display: flex; align-items: center; margin-top: 20px;">  <div data-bbox="671 1128 938 1317" style="margin-left: 10px;"> <p data-bbox="671 1128 767 1158">NOTE</p> <p data-bbox="671 1196 938 1317">If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	
<p data-bbox="164 1964 437 2029">imageContentSources</p>	<p data-bbox="488 1964 858 2029">Sources and repositories for the release-image content.</p>	<p data-bbox="979 1964 1414 2067">Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</p>

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 60px; height: 100px; margin-right: 10px;">  </div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: #ccc; width: 60px; height: 100px; margin-right: 10px;">  </div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

21.4.11.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 21.11. Additional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.rootVolume.size	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
compute.platform.openstack.rootVolume.type	For compute machines, the root volume's type.	String, for example performance .
controlPlane.platform.openstack.rootVolume.size	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
controlPlane.platform.openstack.rootVolume.type	For control plane machines, the root volume's type.	String, for example performance .
platform.openstack.cloud	The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.	String, for example MyCloud .
platform.openstack.externalNetwork	The RHOSP external network name to be used for installation.	String, for example external .
platform.openstack.computeFlavor	<p>The RHOSP flavor to use for control plane and compute machines.</p> <p>This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the type key in the platform.openstack.defaultMachinePlatform property. You can also set a flavor value for each machine pool individually.</p>	String, for example m1.xlarge .

21.4.11.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 21.12. Optional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.additionalNetworkIDs	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .
compute.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .
compute.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
compute.platform.openstack.rootVolume.zones	For compute machines, the availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .

Parameter	Description	Values
compute.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	<p>A server group policy to apply to the machine pool. For example, soft-affinity.</p>
controlPlane.platform.openstack.additionalNetworkIDs	<p>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</p> <p>Additional networks that are attached to a control plane machine are also attached to the bootstrap node.</p>	<p>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</p>
controlPlane.platform.openstack.additionalSecurityGroupIDs	<p>Additional security groups that are associated with control plane machines.</p>	<p>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</p>

Parameter	Description	Values
controlPlane.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
controlPlane.platform.openstack.rootVolume.zones	For control plane machines, the availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .
controlPlane.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations, and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .

Parameter	Description	Values
platform.openstack.clusterOSImage	<p>The location from which the installation program downloads the RHCOS image.</p> <p>You must set this parameter to perform an installation in a restricted network.</p>	<p>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.</p> <p>For example, http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d. The value can also be the name of an existing Glance image, for example my-rhcos.</p>
platform.openstack.clusterOSImageProperties	<p>Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if platform.openstack.clusterOSImage is set to an existing Glance image.</p> <p>You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the hw_scsi_model property value to virtio-scsi and the hw_disk_bus value to scsi.</p> <p>You can also use this property to enable the QEMU guest agent by including the hw_qemu_guest_agent property with a value of yes.</p>	<p>A list of key-value string pairs. For example, ["hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"].</p>
platform.openstack.defaultMachinePlatform	<p>The default machine pool platform configuration.</p>	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</p>	<p>An IP address, for example 128.0.0.1.</p>

Parameter	Description	Values
platform.openstack.apiFloatingIP	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.	An IP address, for example 128.0.0.1 .
platform.openstack.externalDNS	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, ["8.8.8.8", "192.168.1.12"] .
platform.openstack.machinesSubnet	<p>The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</p> <p>The first item in networking.machineNetwork must match the value of machinesSubnet.</p> <p>If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</p>	A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

21.4.11.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

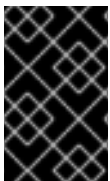
Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.
- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.



NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.



IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

21.4.11.7. Sample customized install-config.yaml file for RHOSP with Kuryr

To deploy with Kuryr SDN instead of the default OVN-Kubernetes network plugin, you must modify the **install-config.yaml** file to include **Kuryr** as the desired **networking.networkType**. This sample **install-config.yaml** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.



IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
```

```
- cidr: 10.0.0.0/16
serviceNetwork:
- 172.30.0.0/16 1
networkType: Kuryr 2
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
    trunkSupport: true 3
    octaviaSupport: true 4
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

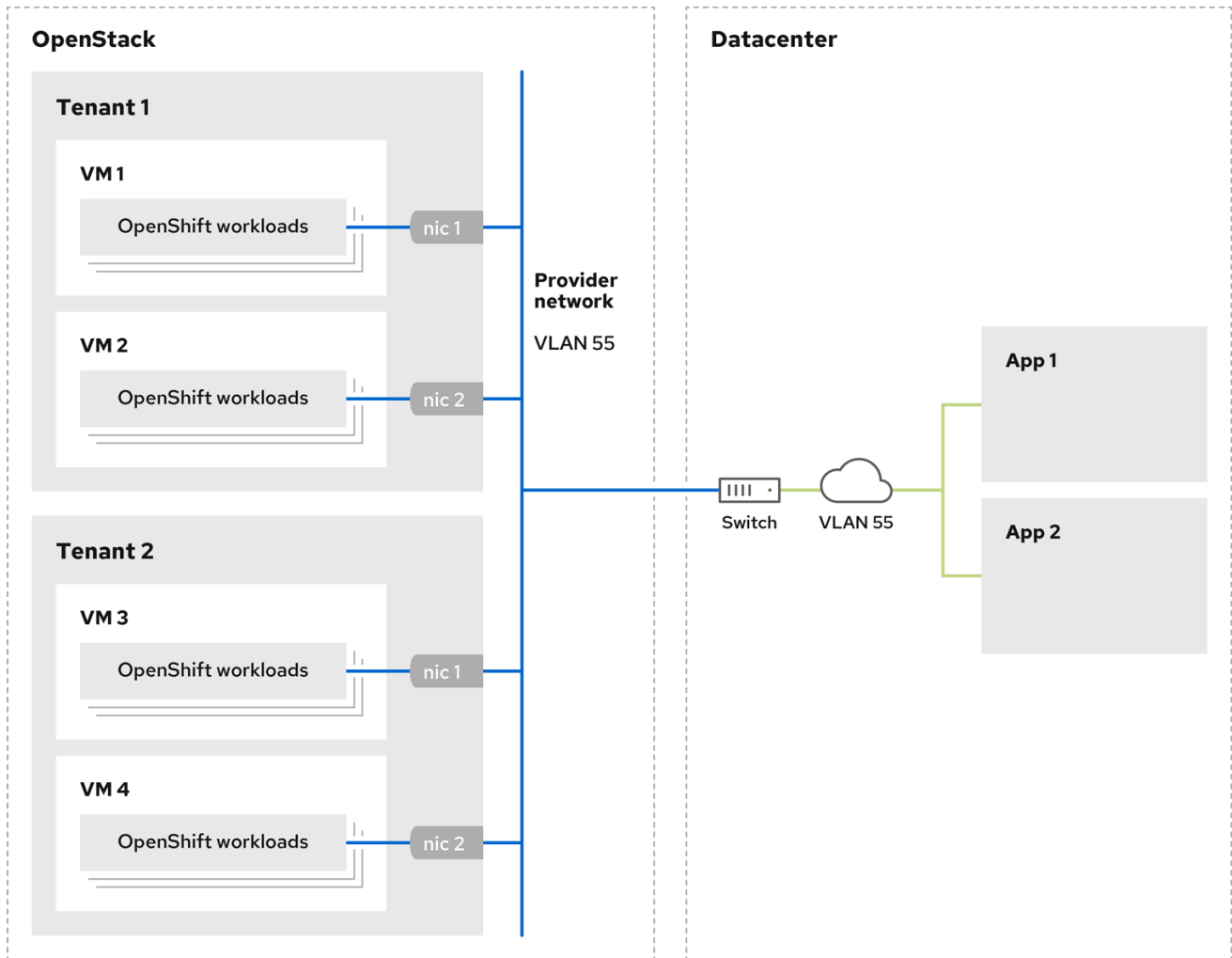
- 1 The Amphora Octavia driver creates two ports per load balancer. As a result, the service subnet that the installer creates is twice the size of the CIDR that is specified as the value of the **serviceNetwork** property. The larger range is required to prevent IP address conflicts.
- 2 The cluster network plugin to install. The supported values are **Kuryr**, **OVNKubernetes**, and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 3 4 Both **trunkSupport** and **octaviaSupport** are automatically discovered by the installer, so there is no need to set them. But if your environment does not meet both requirements, Kuryr SDN will not properly work. Trunks are needed to connect the pods to the RHOSP network and Octavia is required to create the OpenShift Container Platform services.

21.4.11.8. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

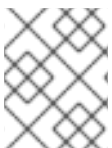
In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



170_OpenShift_0621

OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).



NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in [the RHOSP documentation](#).

21.4.11.8.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The [RHOSP networking service \(Neutron\) is enabled](#) and accessible through the RHOSP networking API.
- The RHOSP networking service has the [port security and allowed address pairs extensions enabled](#).

- The provider network can be shared with other tenants.

TIP

Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

TIP

To create a network for a project that is named "openshift," enter the following command

```
$ openstack network create --project openshift
```

To create a subnet for a project that is named "openshift," enter the following command

```
$ openstack subnet create --project openshift
```

To learn more about creating networks on RHOSP, read [the provider networks documentation](#).

If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.



IMPORTANT

Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.

Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

```
$ openstack subnet create --dhcp --host-route
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- Optional: To secure the network, create [role-based access control \(RBAC\)](#) rules that limit network access to a single project.

21.4.11.8.2. Deploying a cluster that has a primary interface on a provider network

You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.



IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: ①
      - 192.0.2.13
    ingressVIPs: ②
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
  networking:
    machineNetwork:
      - cidr: 192.0.2.0/24
```

- ① ② In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.



WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

TIP

You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see [Understanding multiple networks](#).

21.4.11.9. Kuryr ports pools

A Kuryr ports pool maintains a number of ports on standby for pod creation.

Keeping ports on standby minimizes pod creation time. Without ports pools, Kuryr must explicitly request port creation or deletion whenever a pod is created or deleted.

The Neutron ports that Kuryr uses are created in subnets that are tied to namespaces. These pod ports are also added as subports to the primary port of OpenShift Container Platform cluster nodes.

Because Kuryr keeps each namespace in a separate subnet, a separate ports pool is maintained for each namespace-worker pair.

Prior to installing a cluster, you can set the following parameters in the **cluster-network-03-config.yml** manifest file to configure ports pool behavior:

- The **enablePortPoolsPrepopulation** parameter controls pool prepopulation, which forces Kuryr to add Neutron ports to the pools when the first pod that is configured to use the dedicated network for pods is created in a namespace. The default value is **false**.
- The **poolMinPorts** parameter is the minimum number of free ports that are kept in the pool. The default value is **1**.
- The **poolMaxPorts** parameter is the maximum number of free ports that are kept in the pool. A value of **0** disables that upper bound. This is the default setting.
If your OpenStack port quota is low, or you have a limited number of IP addresses on the pod network, consider setting this option to ensure that unneeded ports are deleted.
- The **poolBatchPorts** parameter defines the maximum number of Neutron ports that can be created at once. The default value is **3**.

21.4.11.10. Adjusting Kuryr ports pools during installation

During installation, you can configure how Kuryr manages Red Hat OpenStack Platform (RHOSP) Neutron ports to control the speed and efficiency of pod creation.

Prerequisites

- Create and modify the **install-config.yaml** file.

Procedure

1. From a command line, create the manifest files:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the name of the directory that contains the **install-config.yaml** file for your cluster.
2. Create a file that is named **cluster-network-03-config.yml** in the `<installation_directory>/manifests/` directory:

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

- 1 For `<installation_directory>`, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-network-*
```

Example output

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. Open the **cluster-network-03-config.yml** file in an editor, and enter a custom resource (CR) that describes the Cluster Network Operator configuration that you want:

```
$ oc edit networks.operator.openshift.io cluster
```

4. Edit the settings to meet your requirements. The following file is provided as an example:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
      enablePortPoolsPrepopulation: false 1
      poolMinPorts: 1 2
      poolBatchPorts: 3 3
      poolMaxPorts: 5 4
      openstackServiceNetwork: 172.30.0.0/15 5
```

- 1 Set **enablePortPoolsPrepopulation** to **true** to make Kuryr create new Neutron ports when the first pod on the network for pods is created in a namespace. This setting raises the Neutron ports quota but can reduce the time that is required to spawn pods. The default

value is **false**.

- 2 Kuryr creates new ports for a pool if the number of free ports in that pool is lower than the value of **poolMinPorts**. The default value is **1**.
- 3 **poolBatchPorts** controls the number of new ports that are created if the number of free ports is lower than the value of **poolMinPorts**. The default value is **3**.
- 4 If the number of free ports in a pool is higher than the value of **poolMaxPorts**, Kuryr deletes them until the number matches that value. Setting this value to **0** disables this upper bound, preventing pools from shrinking. The default value is **0**.
- 5 The **openStackServiceNetwork** parameter defines the CIDR range of the network from which IP addresses are allocated to RHOSP Octavia's LoadBalancers.

If this parameter is used with the Amphora driver, Octavia takes two IP addresses from this network for each load balancer: one for OpenShift and the other for VRRP connections. Because these IP addresses are managed by OpenShift Container Platform and Neutron respectively, they must come from different pools. Therefore, the value of **openStackServiceNetwork** must be at least twice the size of the value of **serviceNetwork**, and the value of **serviceNetwork** must overlap entirely with the range that is defined by **openStackServiceNetwork**.

The CNO verifies that VRRP IP addresses that are taken from the range that is defined by this parameter do not overlap with the range that is defined by the **serviceNetwork** parameter.

If this parameter is not set, the CNO uses an expanded value of **serviceNetwork** that is determined by decrementing the prefix size by 1.

5. Save the **cluster-network-03-config.yml** file, and exit the text editor.
6. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory while creating the cluster.

21.4.12. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

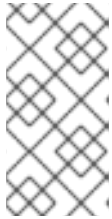
Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

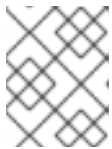
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

■

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

21.4.13. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

21.4.13.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"  
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"  
<external_network>
```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>  
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```



NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your `/etc/hosts` file:

- `<api_floating_ip> api.<cluster_name>.<base_domain>`
- `<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>`
- `application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>`

The cluster domain names in the `/etc/hosts` file grant access to the web console and the monitoring interface of your cluster locally. You can also use the `kubectl` or `oc`. You can access the user applications by using the additional entries pointing to the `<application_floating_ip>`. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

4. Add the FIPs to the `install-config.yaml` file as the values of the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.apiFloatingIP`

If you use these values, you must also enter an external network as the value of the `platform.openstack.externalNetwork` parameter in the `install-config.yaml` file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

21.4.13.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `install-config.yaml` file, do not define the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.apiFloatingIP`

If you cannot provide an external network, you can also leave `platform.openstack.externalNetwork` blank. If you do not provide a value for `platform.openstack.externalNetwork`, a router is not created for

you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

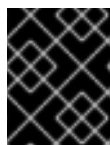
You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

21.4.14. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized `./install-config.yaml` file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

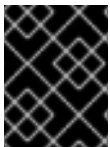
**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

21.4.15. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

- View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

- View your cluster's version:

```
$ oc get clusterversion
```

- View your Operators' status:

```
$ oc get clusteroperator
```

- View all running pods in the cluster:

```
$ oc get pods -A
```

21.4.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

- Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

system:admin

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

21.4.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

21.4.18. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- If you need to enable external access to node ports, [configure ingress cluster traffic by using a node port](#).
- If you did not configure RHOSP to accept application traffic over floating IP addresses, [configure RHOSP access with floating IP addresses](#).

21.5. INSTALLING A CLUSTER ON OPENSTACK ON YOUR OWN INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on Red Hat OpenStack Platform (RHOSP) that runs on user-provisioned infrastructure.

Using your own infrastructure allows you to integrate your cluster with existing infrastructure and modifications. The process requires more labor on your part than installer-provisioned installations, because you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. However, Red Hat provides Ansible playbooks to help you in the deployment process.

21.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

- You verified that OpenShift Container Platform 4.12 is compatible with your RHOSP version by using the [Supported platforms for OpenShift clusters](#) section. You can also compare platform support across different versions by viewing the [OpenShift Container Platform on RHOSP support matrix](#).
- You have an RHOSP account where you want to install OpenShift Container Platform.
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see [Recommended practices for scaling the cluster](#).
- On the machine from which you run the installation program, you have:
 - A single directory in which you can keep the files you create during the installation process
 - Python 3

21.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

21.5.3. Resource guidelines for installing OpenShift Container Platform on RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 21.13. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

Resource	Value
Floating IP addresses	3
Ports	15
Routers	1

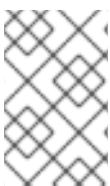
Resource	Value
Subnets	1
RAM	88 GB
vCPUs	22
Volume storage	275 GB
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

21.5.3.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs

- At least 100 GB storage space from the RHOSP quota

21.5.3.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

21.5.3.3. Bootstrap machine

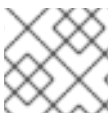
During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.5.4. Downloading playbook dependencies

The Ansible playbooks that simplify the installation process on user-provisioned infrastructure require several Python modules. On the machine where you will run the installer, add the modules' repositories and then download them.



NOTE

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:
 - a. Register with Red Hat Subscription Manager:

```
$ sudo subscription-manager register # If not done already
```

- b. Pull the latest subscription data:

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

- c. Disable the current repositories:

```
$ sudo subscription-manager repos --disable=* # If not done already
```

- d. Add the required repositories:

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. Install the modules:

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
ansible-collections-openstack
```

3. Ensure that the **python** command points to **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

21.5.5. Downloading the installation playbooks

Download Ansible playbooks that you can use to install OpenShift Container Platform on your own Red Hat OpenStack Platform (RHOSP) infrastructure.

Prerequisites

- The curl command-line tool is available on your machine.

Procedure

- To download the playbooks to your working directory, run the following script from a command line:

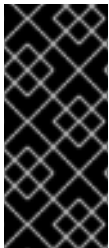
```
$ xargs -n 1 curl -O <<< '
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/bootstrap.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/common.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/compute-nodes.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/control-
plane.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/inventory.yaml
```

```

https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/network.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
load-balancers.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
containers.yaml'

```

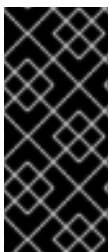
The playbooks are downloaded to your machine.



IMPORTANT

During the installation process, you can modify the playbooks to configure your deployment.

Retain all playbooks for the life of your cluster. You must have the playbooks to remove your OpenShift Container Platform cluster from RHOSP.



IMPORTANT

You must match any edits you make in the **bootstrap.yaml**, **compute-nodes.yaml**, **control-plane.yaml**, **network.yaml**, and **security-groups.yaml** files to the corresponding playbooks that are prefixed with **down-**. For example, edits to the **bootstrap.yaml** file must be reflected in the **down-bootstrap.yaml** file, too. If you do not edit both files, the supported cluster removal process will fail.

21.5.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

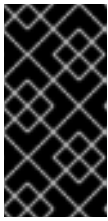
Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

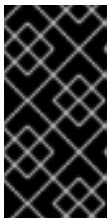
1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

21.5.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

■

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

21.5.8. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image

The OpenShift Container Platform installation program requires that a Red Hat Enterprise Linux CoreOS (RHCOS) image be present in the Red Hat OpenStack Platform (RHOSP) cluster. Retrieve the latest RHCOS image, then upload it using the RHOSP CLI.

Prerequisites

- The RHOSP CLI is installed.

Procedure

1. Log in to the Red Hat Customer Portal's [Product Downloads page](#).
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.12 for Red Hat Enterprise Linux (RHEL) 8.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the *Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)*.
4. Decompress the image.



NOTE

You must decompress the RHOSP image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like **.gz** or **.tgz**. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

- From the image that you downloaded, create an image that is named **rhcos** in your cluster by using the RHOSP CLI:

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-
${RHCOS_VERSION}-openstack.qcow2 rhcos
```



IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either **.raw** or **.qcow2** formats. If you use Ceph, you must use the **.raw** format.



WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

After you upload the image to RHOSP, it is usable in the installation process.

21.5.9. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries

Procedure

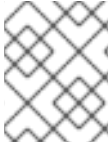
- Using the RHOSP CLI, verify the name and ID of the 'External' network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+-----+-----+-----+
| ID                | Name          | Router Type |
+-----+-----+-----+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+-----+-----+-----+
```

A network with an external router type appears in the network list. If at least one does not, see [Creating a default floating IP network](#) and [Creating a default provider network](#).

**NOTE**

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see [Neutron trunk port](#).

21.5.10. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

21.5.10.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API, cluster applications, and the bootstrap process.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. By using the Red Hat OpenStack Platform (RHOSP) CLI, create the bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```



NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your **/etc/hosts** file:

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

The cluster domain names in the **/etc/hosts** file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the **<application_floating_ip>**. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

5. Add the FIPs to the **inventory.yaml** file as the values of the following variables:

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

If you use these values, you must also enter an external network as the value of the **os_external_network** variable in the **inventory.yaml** file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

21.5.10.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the **inventory.yaml** file, do not define the following variables:

- **os_api_fip**
- **os_bootstrap_fip**

- **os_ingress_fip**

If you cannot provide an external network, you can also leave **os_external_network** blank. If you do not provide a value for **os_external_network**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. Later in the installation process, when you create network resources, you must configure external connectivity on your own.

If you run the installer with the **wait-for** command from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

21.5.11. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:
 - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.



IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in [a separate file](#) from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see [Config files](#) in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
```

```
dev-env:
  region_name: RegionOne
  auth:
    username: <username>
    password: <password>
    project_name: 'devonly'
    auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
 - a. Copy the certificate authority file to your machine.
 - b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
  ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
 - a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
 - b. The current directory
 - c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**
 - d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**
The installation program searches for **clouds.yaml** in that order.

21.5.12. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **openstack** as the platform to target.
 - iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.
 - iv. Specify the floating IP address to use for external access to the OpenShift API.
 - v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.
 - vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.
 - vii. Enter a name for your cluster. The name must be 14 or fewer characters long.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

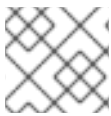
**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

You now have the file **install-config.yaml** in the directory that you specified.

21.5.13. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

21.5.13.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 21.14. Required parameters

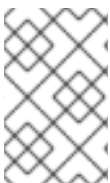
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>. <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev . The string must be 14 characters or fewer long.
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform , consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

21.5.13.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 21.15. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

21.5.13.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 21.16. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}

Parameter	Description	Values
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div data-bbox="485 1473 593 1792" data-label="Image"> </div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 584 592 1391" style="background-color: black; color: white; padding: 5px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="488 1435 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...

21.5.13.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 21.17. Additional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.rootVolume.size	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .

Parameter	Description	Values
compute.platform.openstack.rootVolume.type	For compute machines, the root volume's type.	String, for example performance .
controlPlane.platform.openstack.rootVolume.size	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
controlPlane.platform.openstack.rootVolume.type	For control plane machines, the root volume's type.	String, for example performance .
platform.openstack.cloud	The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.	String, for example MyCloud .
platform.openstack.externalNetwork	The RHOSP external network name to be used for installation.	String, for example external .
platform.openstack.computeFlavor	<p>The RHOSP flavor to use for control plane and compute machines.</p> <p>This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the type key in the platform.openstack.defaultMachinePlatform property. You can also set a flavor value for each machine pool individually.</p>	String, for example m1.xlarge .

21.5.13.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 21.18. Optional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.additionalNetworkIDs	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .
compute.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .
compute.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
compute.platform.openstack.rootVolume.zones	For compute machines, the availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .

Parameter	Description	Values
compute.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	<p>A server group policy to apply to the machine pool. For example, soft-affinity.</p>
controlPlane.platform.openstack.additionalNetworkIDs	<p>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</p> <p>Additional networks that are attached to a control plane machine are also attached to the bootstrap node.</p>	<p>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</p>
controlPlane.platform.openstack.additionalSecurityGroupIDs	<p>Additional security groups that are associated with control plane machines.</p>	<p>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</p>

Parameter	Description	Values
controlPlane.placement.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
controlPlane.placement.openstack.rootVolume.zones	For control plane machines, the availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .
controlPlane.placement.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations, and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .

Parameter	Description	Values
platform.openstack.clusterOSImage	<p>The location from which the installation program downloads the RHCOS image.</p> <p>You must set this parameter to perform an installation in a restricted network.</p>	<p>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.</p> <p>For example, http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d. The value can also be the name of an existing Glance image, for example my-rhcos.</p>
platform.openstack.clusterOSImageProperties	<p>Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if platform.openstack.clusterOSImage is set to an existing Glance image.</p> <p>You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the hw_scsi_model property value to virtio-scsi and the hw_disk_bus value to scsi.</p> <p>You can also use this property to enable the QEMU guest agent by including the hw_qemu_guest_agent property with a value of yes.</p>	<p>A list of key-value string pairs. For example, [{"hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"}].</p>
platform.openstack.defaultMachinePlatform	<p>The default machine pool platform configuration.</p>	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</p>	<p>An IP address, for example 128.0.0.1.</p>

Parameter	Description	Values
platform.openstack.apiFloatingIP	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.	An IP address, for example 128.0.0.1 .
platform.openstack.externalDNS	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, ["8.8.8.8", "192.168.1.12"] .
platform.openstack.machinesSubnet	<p>The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</p> <p>The first item in networking.machineNetwork must match the value of machinesSubnet.</p> <p>If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</p>	A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

21.5.13.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.
- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.



NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.



IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

21.5.13.7. Sample customized **install-config.yaml** file for RHOSP

This sample **install-config.yaml** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.



IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
    replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
```

```
- 172.30.0.0/16
networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

21.5.13.8. Setting a custom subnet for machines

The IP range that the installation program uses by default might not match the Neutron subnet that you create when you install OpenShift Container Platform. If necessary, update the CIDR value for new machines by editing the installation configuration file.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.

Procedure

- On a command line, browse to the directory that contains **install-config.yaml**.
- From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:
 - To set the value by using a script, run:

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; 1
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- 1** Insert a value that matches your intended Neutron subnet, e.g. **192.0.2.0/24**.

- To set the value manually, open the file and set the value of **networking.machineCIDR** to something that matches your intended Neutron subnet.

21.5.13.9. Emptying compute machine pools

To proceed with an installation that uses your own infrastructure, set the number of compute machines in the installation configuration file to zero. Later, you create these machines manually.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.

Procedure

1. On a command line, browse to the directory that contains **install-config.yaml**.
2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:
 - To set the value by using a script, run:

```
$ python -c '  
import yaml;  
path = "install-config.yaml";  
data = yaml.safe_load(open(path));  
data["compute"][0]["replicas"] = 0;  
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

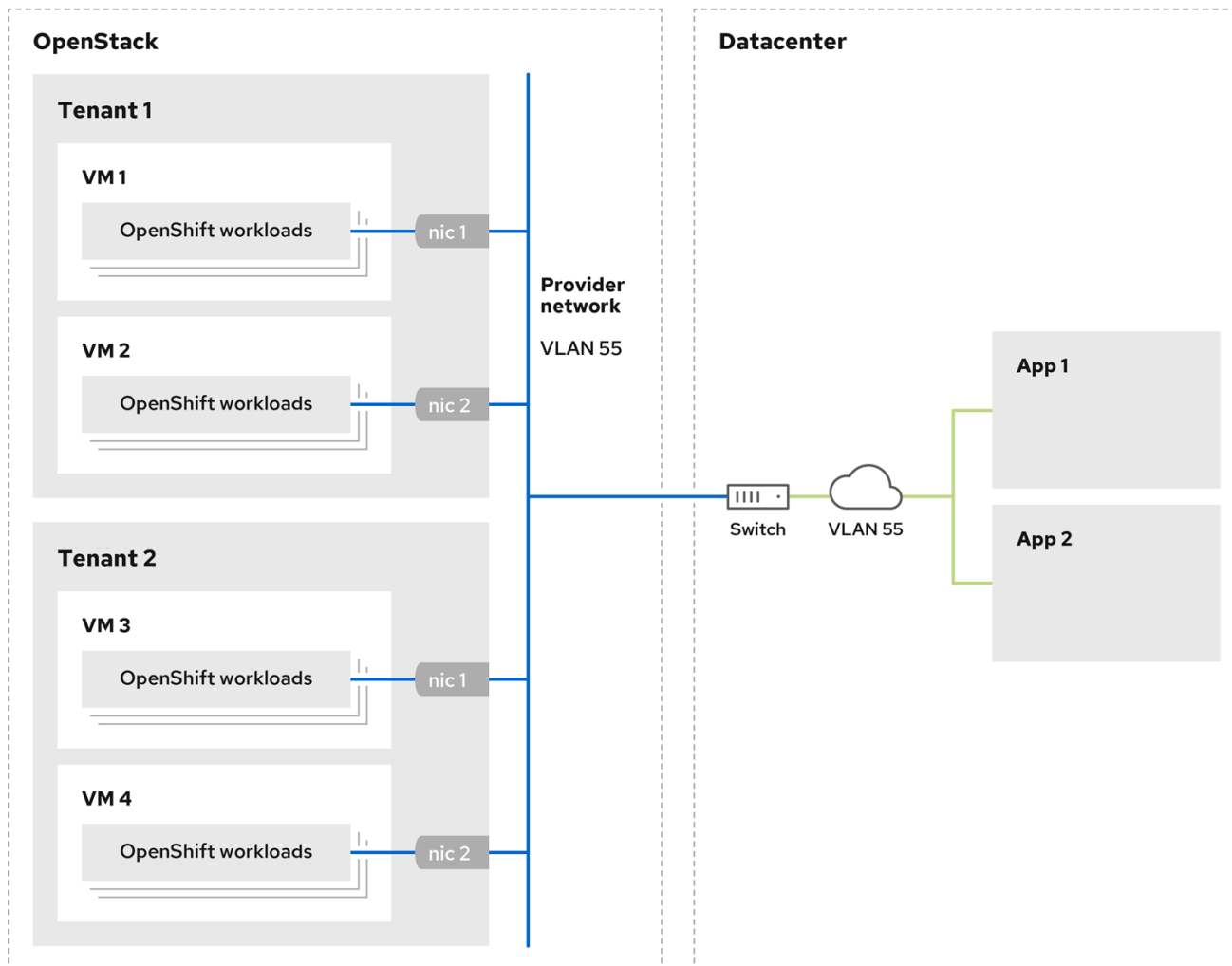
- To set the value manually, open the file and set the value of **compute.<first entry>.replicas** to **0**.

21.5.13.10. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



170_OpenShift_0621

OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).



NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in [the RHOSP documentation](#).

21.5.13.10.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The [RHOSP networking service \(Neutron\) is enabled](#) and accessible through the RHOSP networking API.
- The RHOSP networking service has the [port security and allowed address pairs extensions enabled](#).

- The provider network can be shared with other tenants.

TIP

Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

TIP

To create a network for a project that is named "openshift," enter the following command

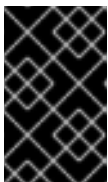
```
$ openstack network create --project openshift
```

To create a subnet for a project that is named "openshift," enter the following command

```
$ openstack subnet create --project openshift
```

To learn more about creating networks on RHOSP, read [the provider networks documentation](#).

If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.



IMPORTANT

Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.

Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

```
$ openstack subnet create --dhcp --host-route
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- Optional: To secure the network, create [role-based access control \(RBAC\)](#) rules that limit network access to a single project.

21.5.13.10.2. Deploying a cluster that has a primary interface on a provider network

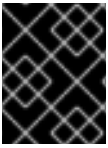
You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.



IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: 1
    - 192.0.2.13
    ingressVIPs: 2
    - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
  networking:
    machineNetwork:
      - cidr: 192.0.2.0/24
```

- 1 2** In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.



WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

TIP

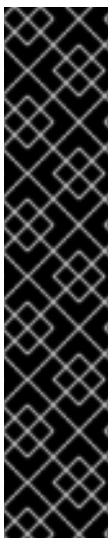
You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see [Understanding multiple networks](#).

21.5.14. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.

**IMPORTANT**

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
3. Check that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
 4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1** For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5. Export the metadata file's **infraID** key as an environment variable:

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

TIP

Extract the **infraID** key from **metadata.json** and use it as a prefix for all of the RHOSP resources that you create. By doing so, you avoid name conflicts when making multiple deployments in the same project.

21.5.15. Preparing the bootstrap Ignition files

The OpenShift Container Platform installation process relies on bootstrap machines that are created from a bootstrap Ignition configuration file.

Edit the file and upload it. Then, create a secondary bootstrap Ignition configuration file that Red Hat OpenStack Platform (RHOSP) uses to download the primary file.

Prerequisites

- You have the bootstrap Ignition file that the installer program generates, **bootstrap.ign**.
- The infrastructure ID from the installer's metadata file is set as an environment variable (**\$INFRA_ID**).
 - If the variable is not set, see **Creating the Kubernetes manifest and Ignition config files**
- You have an HTTP(S)-accessible way to store the bootstrap Ignition file.
 - The documented procedure uses the RHOSP image service (Glance), but you can also use the RHOSP storage service (Swift), Amazon S3, an internal HTTP server, or an ad hoc Nova server.

Procedure

1. Run the following Python script. The script modifies the bootstrap Ignition file to set the hostname and, if available, CA certificate file when it runs:

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
    {
        'path': '/etc/hostname',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
        }
    }
)

ca_cert_path = os.environ.get('OS_CACERT', "")
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    }
)

ignition['storage']['files'] = files;
```

```
with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

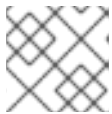
- Using the RHOSP CLI, create an image that uses the bootstrap Ignition file:

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>
```

- Get the image's details:

```
$ openstack image show <image_name>
```

Make a note of the **file** value; it follows the pattern **v2/images/<image_ID>/file**.



NOTE

Verify that the image you created is active.

- Retrieve the image service's public address:

```
$ openstack catalog show image
```

- Combine the public address with the image **file** value and save the result as the storage location. The location follows the pattern **<image_service_public_URL>/v2/images/<image_ID>/file**.

- Generate an auth token and save the token ID:

```
$ openstack token issue -c id -f value
```

- Insert the following content into a file called **\$INFRA_ID-bootstrap-ignition.json** and edit the placeholders to match your own values:

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
        }]
      }
    }
  },
}
```

```

"version": "3.2.0"
}
}

```

- 1 Replace the value of **ignition.config.merge.source** with the bootstrap Ignition file storage URL.
- 2 Set **name** in **httpHeaders** to **"X-Auth-Token"**.
- 3 Set **value** in **httpHeaders** to your token's ID.
- 4 If the bootstrap Ignition file server uses a self-signed certificate, include the base64-encoded certificate.

8. Save the secondary Ignition config file.

The bootstrap Ignition data will be passed to RHOSP during installation.



WARNING

The bootstrap Ignition file contains sensitive information, like **clouds.yaml** credentials. Ensure that you store it in a secure place, and delete it after you complete the installation process.

21.5.16. Creating control plane Ignition config files on RHOSP

Installing OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) on your own infrastructure requires control plane Ignition config files. You must create multiple config files.



NOTE

As with the bootstrap Ignition configuration, you must explicitly define a hostname for each control plane machine.

Prerequisites

- The infrastructure ID from the installation program's metadata file is set as an environment variable (**\$INFRA_ID**).
 - If the variable is not set, see "Creating the Kubernetes manifest and Ignition config files".

Procedure

- On a command line, run the following Python script:

```

$ for index in $(seq 0 2); do
  MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
  python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});

```

```

files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
json.dump(ignition, sys.stdout) <master.ign >"$INFRA_ID-master-$index-ignition.json"
done

```

You now have three control plane Ignition files: **<INFRA_ID>-master-0-ignition.json**, **<INFRA_ID>-master-1-ignition.json**, and **<INFRA_ID>-master-2-ignition.json**.

21.5.17. Creating network resources on RHOSP

Create the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires. To save time, run supplied Ansible playbooks that generate security groups, networks, subnets, routers, and ports.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".

Procedure

1. Optional: Add an external network value to the **inventory.yaml** playbook:

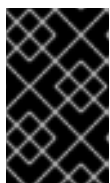
Example external network value in the **inventory.yaml** Ansible playbook

```

...
# The public network providing connectivity to the cluster. If not
# provided, the cluster external connectivity must be provided in another
# way.

# Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
os_external_network: 'external'
...

```



IMPORTANT

If you did not provide a value for **os_external_network** in the **inventory.yaml** file, you must ensure that VMs can access Glance and an external connection yourself.

2. Optional: Add external network and floating IP (FIP) address values to the **inventory.yaml** playbook:

Example FIP values in the **inventory.yaml** Ansible playbook

```

...

```

```

# OpenShift API floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the Control Plane to
# serve the OpenShift API.
os_api_fip: '203.0.113.23'

# OpenShift Ingress floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the worker nodes to serve
# the applications.
os_ingress_fip: '203.0.113.19'

# If this value is non-empty, the corresponding floating IP will be
# attached to the bootstrap machine. This is needed for collecting logs
# in case of install failure.
os_bootstrap_fip: '203.0.113.20'

```



IMPORTANT

If you do not define values for **os_api_fip** and **os_ingress_fip**, you must perform postinstallation network configuration.

If you do not define a value for **os_bootstrap_fip**, the installer cannot download debugging information from failed installations.

See "Enabling access to the environment" for more information.

3. On a command line, create security groups by running the **security-groups.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. On a command line, create a network, subnet, and router by running the **network.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml network.yaml
```

5. Optional: If you want to control the default resolvers that Nova servers use, run the RHOSP CLI command:

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

Optionally, you can use the **inventory.yaml** file that you created to customize your installation. For example, you can deploy a cluster that uses bare metal machines.

21.5.17.1. Deploying a cluster with bare metal machines

If you want your cluster to use bare metal machines, modify the **inventory.yaml** file. Your cluster can have both control plane and compute machines running on bare metal, or just compute machines.

Bare-metal compute machines are not supported on clusters that use Kuryr.



NOTE

Be sure that your **install-config.yaml** file reflects whether the RHOSP network that you use for bare metal workers supports floating IP addresses or not.

Prerequisites

- The RHOSP [Bare Metal service \(Ironic\)](#) is enabled and accessible via the RHOSP Compute API.
- Bare metal is available as [a RHOSP flavor](#).
- If your cluster runs on an RHOSP version that is more than 16.1.6 and less than 16.2.4, bare metal workers do not function due to a [known issue](#) that causes the metadata service to be unavailable for services on OpenShift Container Platform nodes.
- The RHOSP network supports both VM and bare metal server attachment.
- Your network configuration does not rely on a provider network. Provider networks are not supported.
- If you want to deploy the machines on a pre-existing network, a RHOSP subnet is provisioned.
- If you want to deploy the machines on an installer-provisioned network, the RHOSP Bare Metal service (Ironic) is able to listen for and interact with Preboot eXecution Environment (PXE) boot machines that run on tenant networks.
- You created an **inventory.yaml** file as part of the OpenShift Container Platform installation process.

Procedure

1. In the **inventory.yaml** file, edit the flavors for machines:
 - a. If you want to use bare-metal control plane machines, change the value of **os_flavor_master** to a bare metal flavor.
 - b. Change the value of **os_flavor_worker** to a bare metal flavor.

An example bare metal inventory.yaml file

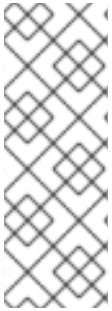
```
all:
  hosts:
    localhost:
      ansible_connection: local
      ansible_python_interpreter: "{{ansible_playbook_python}}"

      # User-provided values
      os_subnet_range: '10.0.0.0/16'
      os_flavor_master: 'my-bare-metal-flavor' 1
      os_flavor_worker: 'my-bare-metal-flavor' 2
      os_image_rhcos: 'rhcos'
      os_external_network: 'external'
  ...
```

- 1 If you want to have bare-metal control plane machines, change this value to a bare metal flavor.

- 2 Change this value to a bare metal flavor to use for compute machines.

Use the updated **inventory.yaml** file to complete the installation process. Machines that are created during deployment use the flavor that you added to the file.



NOTE

The installer may time out while waiting for bare metal machines to boot.

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

21.5.18. Creating the bootstrap machine on RHOSP

Create a bootstrap machine and give it the network access it needs to run on Red Hat OpenStack Platform (RHOSP). Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **bootstrap.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **bootstrap.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. After the bootstrap server is active, view the logs to verify that the Ignition files were received:

```
$ openstack console log show "$INFRA_ID-bootstrap"
```

21.5.19. Creating the control plane machines on RHOSP

Create three control plane machines by using the Ignition config files that you generated. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".

- You downloaded the playbooks in "Downloading the installation playbooks".
- The infrastructure ID from the installation program's metadata file is set as an environment variable (**\$INFRA_ID**).
- The **inventory.yaml**, **common.yaml**, and **control-plane.yaml** Ansible playbooks are in a common directory.
- You have the three Ignition files that were created in "Creating control plane Ignition config files".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. If the control plane Ignition config files aren't already in your working directory, copy them into it.
3. On a command line, run the **control-plane.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. Run the following command to monitor the bootstrapping process:

```
$ openshift-install wait-for bootstrap-complete
```

You will see messages that confirm that the control plane machines are running and have joined the cluster:

```
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

21.5.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```


- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

21.5.21. Deleting bootstrap resources from RHOSP

Delete the bootstrap resources that you no longer need.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **down-bootstrap.yaml** Ansible playbooks are in a common directory.
- The control plane machines are running.
 - If you do not know the status of the machines, see "Verifying cluster status".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **down-bootstrap.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

The bootstrap port, server, and floating IP address are deleted.



WARNING

If you did not disable the bootstrap Ignition file URL earlier, do so now.

21.5.22. Creating compute machines on RHOSP

After standing up the control plane, create compute machines. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **compute-nodes.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.
- The control plane is active.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

Next steps

- Approve the certificate signing requests for the machines.

21.5.23. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE    VERSION
master-0  Ready    master   63m    v1.25.0
master-1  Ready    master   63m    v1.25.0
master-2  Ready    master   64m    v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME          AGE   REQUESTOR                                     CONDITION
csr-8b2br    15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps    15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

21.5.24. Verifying a successful installation

Verify that the OpenShift Container Platform installation is complete.

Prerequisites

- You have the installation program (**openshift-install**)

Procedure

- On a command line, enter:

```
$ openshift-install --log-level debug wait-for install-complete
```

The program outputs the console URL, as well as the administrator's login information.

21.5.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

21.5.26. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- If you need to enable external access to node ports, [configure ingress cluster traffic by using a node port](#).
- If you did not configure RHOSP to accept application traffic over floating IP addresses, [configure RHOSP access with floating IP addresses](#).

21.6. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR ON YOUR OWN INFRASTRUCTURE



IMPORTANT

Kuryr is a deprecated feature. Deprecated functionality is still included in OpenShift Container Platform and continues to be supported; however, it will be removed in a future release of this product and is not recommended for new deployments.

For the most recent list of major functionality that has been deprecated or removed within OpenShift Container Platform, refer to the *Deprecated and removed features* section of the OpenShift Container Platform release notes.

In OpenShift Container Platform version 4.12, you can install a cluster on Red Hat OpenStack Platform (RHOSP) that runs on user-provisioned infrastructure.

Using your own infrastructure allows you to integrate your cluster with existing infrastructure and modifications. The process requires more labor on your part than installer-provisioned installations, because you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. However, Red Hat provides Ansible playbooks to help you in the deployment process.

21.6.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You verified that OpenShift Container Platform 4.12 is compatible with your RHOSP version by using the [Supported platforms for OpenShift clusters](#) section. You can also compare platform support across different versions by viewing the [OpenShift Container Platform on RHOSP support matrix](#).
- You have an RHOSP account where you want to install OpenShift Container Platform.
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see [Recommended practices for scaling the cluster](#).
- On the machine from which you run the installation program, you have:
 - A single directory in which you can keep the files you create during the installation process
 - Python 3

21.6.2. About Kuryr SDN



IMPORTANT

Kuryr is a deprecated feature. Deprecated functionality is still included in OpenShift Container Platform and continues to be supported; however, it will be removed in a future release of this product and is not recommended for new deployments.

For the most recent list of major functionality that has been deprecated or removed within OpenShift Container Platform, refer to the *Deprecated and removed features* section of the OpenShift Container Platform release notes.

[Kuryr](#) is a container network interface (CNI) plugin solution that uses the [Neutron](#) and [Octavia](#) Red Hat OpenStack Platform (RHOSP) services to provide networking for pods and Services.

Kuryr and OpenShift Container Platform integration is primarily designed for OpenShift Container Platform clusters running on RHOSP VMs. Kuryr improves the network performance by plugging OpenShift Container Platform pods into RHOSP SDN. In addition, it provides interconnectivity between pods and RHOSP virtual instances.

Kuryr components are installed as pods in OpenShift Container Platform using the **openshift-kuryr** namespace:

- **kuryr-controller** - a single service instance installed on a **master** node. This is modeled in OpenShift Container Platform as a **Deployment** object.
- **kuryr-cni** - a container installing and configuring Kuryr as a CNI driver on each OpenShift Container Platform node. This is modeled in OpenShift Container Platform as a **DaemonSet** object.

The Kuryr controller watches the OpenShift Container Platform API server for pod, service, and namespace create, update, and delete events. It maps the OpenShift Container Platform API calls to corresponding objects in Neutron and Octavia. This means that every network solution that implements the Neutron trunk port functionality can be used to back OpenShift Container Platform via Kuryr. This includes open source solutions such as Open vSwitch (OVS) and Open Virtual Network (OVN) as well as Neutron-compatible commercial SDNs.

Kuryr is recommended for OpenShift Container Platform deployments on encapsulated RHOSP tenant networks to avoid double encapsulation, such as running an encapsulated OpenShift Container Platform SDN over an RHOSP network.

If you use provider networks or tenant VLANs, you do not need to use Kuryr to avoid double encapsulation. The performance benefit is negligible. Depending on your configuration, though, using Kuryr to avoid having two overlays might still be beneficial.

Kuryr is not recommended in deployments where all of the following criteria are true:

- The RHOSP version is less than 16.
- The deployment uses UDP services, or a large number of TCP services on few hypervisors.

or

- The **ovn-octavia** Octavia driver is disabled.
- The deployment uses a large number of TCP services on few hypervisors.

21.6.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr

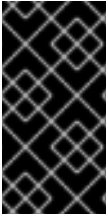
When using Kuryr SDN, the pods, services, namespaces, and network policies are using resources from the RHOSP quota; this increases the minimum requirements. Kuryr also has some additional requirements on top of what a default install requires.

Use the following quota to satisfy a default cluster's minimum requirements:

Table 21.19. Recommended resources for a default OpenShift Container Platform cluster on RHOSP with Kuryr

Resource	Value
Floating IP addresses	3 - plus the expected number of Services of LoadBalancer type
Ports	1500 - 1 needed per Pod
Routers	1
Subnets	250 - 1 needed per Namespace/Project
Networks	250 - 1 needed per Namespace/Project
RAM	112 GB
vCPUs	28
Volume storage	275 GB
Instances	7
Security groups	250 - 1 needed per Service and per NetworkPolicy
Security group rules	1000
Server groups	2 - plus 1 for each additional availability zone in each machine pool
Load balancers	100 - 1 needed per Service
Load balancer listeners	500 - 1 needed per Service-exposed port
Load balancer pools	500 - 1 needed per Service-exposed port

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



IMPORTANT

If you are using Red Hat OpenStack Platform (RHOSP) version 16 with the Amphora driver rather than the OVN Octavia driver, security groups are associated with service accounts instead of user projects.

Take the following notes into consideration when setting resources:

- The number of ports that are required is larger than the number of pods. Kuryr uses ports pools to have pre-created ports ready to be used by pods and speed up the pods' booting time.
- Each network policy is mapped into an RHOSP security group, and depending on the **NetworkPolicy** spec, one or more rules are added to the security group.
- Each service is mapped to an RHOSP load balancer. Consider this requirement when estimating the number of security groups required for the quota.
If you are using RHOSP version 15 or earlier, or the **ovn-octavia driver**, each load balancer has a security group with the user project.
- The quota does not account for load balancer resources (such as VM resources), but you must consider these resources when you decide the RHOSP deployment's size. The default installation will have more than 50 load balancers; the clusters must be able to accommodate them.
If you are using RHOSP version 16 with the OVN Octavia driver enabled, only one load balancer VM is generated; services are load balanced through OVN flows.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

To enable Kuryr SDN, your environment must meet the following requirements:

- Run RHOSP 13+.
- Have Overcloud with Octavia.
- Use Neutron Trunk ports extension.
- Use **openvswitch** firewall driver if ML2/OVS Neutron driver is used instead of **ovs-hybrid**.

21.6.3.1. Increasing quota

When using Kuryr SDN, you must increase quotas to satisfy the Red Hat OpenStack Platform (RHOSP) resources used by pods, services, namespaces, and network policies.

Procedure

- Increase the quotas for a project by running the following command:

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets
250 --networks 250 <project>
```

21.6.3.2. Configuring Neutron

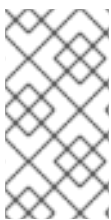
Kuryr CNI leverages the Neutron Trunks extension to plug containers into the Red Hat OpenStack Platform (RHOSP) SDN, so you must use the **trunks** extension for Kuryr to properly work.

In addition, if you leverage the default ML2/OVS Neutron driver, the firewall must be set to **opendvswitch** instead of **ovs_hybrid** so that security groups are enforced on trunk subports and Kuryr can properly handle network policies.

21.6.3.3. Configuring Octavia

Kuryr SDN uses Red Hat OpenStack Platform (RHOSP)'s Octavia LBaaS to implement OpenShift Container Platform services. Thus, you must install and configure Octavia components in RHOSP to use Kuryr SDN.

To enable Octavia, you must include the Octavia service during the installation of the RHOSP Overcloud, or upgrade the Octavia service if the Overcloud already exists. The following steps for enabling Octavia apply to both a clean install of the Overcloud or an Overcloud update.



NOTE

The following steps only capture the key pieces required during the [deployment of RHOSP](#) when dealing with Octavia. It is also important to note that [registry methods](#) vary.

This example uses the local registry method.

Procedure

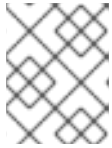
1. If you are using the local registry, create a template to upload the images to the registry. For example:

```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
--tag-from-label {version}-{product-version} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

2. Verify that the **local_registry_images.yaml** file contains the Octavia images. For example:

```
...
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
```

```
push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
push_destination: <local-ip-from-undercloud.conf>:8787
```

**NOTE**

The Octavia container versions vary depending upon the specific RHOSP release installed.

3. Pull the container images from **registry.redhat.io** to the Undercloud node:

```
(undercloud) $ sudo openstack overcloud container image upload \
--config-file /home/stack/local_registry_images.yaml \
--verbose
```

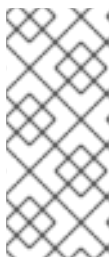
This may take some time depending on the speed of your network and Undercloud disk.

4. Install or update your Overcloud environment with Octavia:

```
$ openstack overcloud deploy --templates \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
-e octavia_timeouts.yaml
```

**NOTE**

This command only includes the files associated with Octavia; it varies based on your specific installation of RHOSP. See the RHOSP documentation for further information. For more information on customizing your Octavia installation, see [installation of Octavia using Director](#).

**NOTE**

When leveraging Kuryr SDN, the Overcloud installation requires the Neutron **trunk** extension. This is available by default on director deployments. Use the **openvswitch** firewall instead of the default **ovs-hybrid** when the Neutron backend is ML2/OVS. There is no need for modifications if the backend is ML2/OVN.

21.6.3.3.1. The Octavia OVN Driver

Octavia supports multiple provider drivers through the Octavia API.

To see all available Octavia provider drivers, on a command line, enter:

```
$ openstack loadbalancer provider list
```

Example output

```
+-----+-----+
| name   | description                               |
+-----+-----+
| amphora | The Octavia Amphora driver.              |
```

octavia	Deprecated alias of the Octavia Amphora driver.
ovn	Octavia OVN driver.
+-----+	

Beginning with RHOSP version 16, the Octavia OVN provider driver (**ovn**) is supported on OpenShift Container Platform on RHOSP deployments.

ovn is an integration driver for the load balancing that Octavia and OVN provide. It supports basic load balancing capabilities, and is based on OpenFlow rules. The driver is automatically enabled in Octavia by Director on deployments that use OVN Neutron ML2.

The Amphora provider driver is the default driver. If **ovn** is enabled, however, Kuryr uses it.

If Kuryr uses **ovn** instead of Amphora, it offers the following benefits:

- Decreased resource requirements. Kuryr does not require a load balancer VM for each service.
- Reduced network latency.
- Increased service creation speed by using OpenFlow rules instead of a VM for each service.
- Distributed load balancing actions across all nodes instead of centralized on Amphora VMs.

21.6.3.4. Known limitations of installing with Kuryr

Using OpenShift Container Platform with Kuryr SDN has several known limitations.

RHOSP general limitations

Using OpenShift Container Platform with Kuryr SDN has several limitations that apply to all versions and environments:

- **Service** objects with the **NodePort** type are not supported.
- Clusters that use the OVN Octavia provider driver support **Service** objects for which the **.spec.selector** property is unspecified only if the **.subsets.addresses** property of the **Endpoints** object includes the subnet of the nodes or pods.
- If the subnet on which machines are created is not connected to a router, or if the subnet is connected, but the router has no external gateway set, Kuryr cannot create floating IPs for **Service** objects with type **LoadBalancer**.
- Configuring the **sessionAffinity=ClientIP** property on **Service** objects does not have an effect. Kuryr does not support this setting.

RHOSP version limitations

Using OpenShift Container Platform with Kuryr SDN has several limitations that depend on the RHOSP version.

- RHOSP versions before 16 use the default Octavia load balancer driver (Amphora). This driver requires that one Amphora load balancer VM is deployed per OpenShift Container Platform service. Creating too many services can cause you to run out of resources. Deployments of later versions of RHOSP that have the OVN Octavia driver disabled also use the Amphora driver. They are subject to the same resource concerns as earlier versions of RHOSP.
- Kuryr SDN does not support automatic unidling by a service.

RHOSP upgrade limitations

As a result of the RHOSP upgrade process, the Octavia API might be changed, and upgrades to the Amphora images that are used for load balancers might be required.

You can address API changes on an individual basis.

If the Amphora image is upgraded, the RHOSP operator can handle existing load balancer VMs in two ways:

- Upgrade each VM by triggering a [load balancer failover](#).
- Leave responsibility for upgrading the VMs to users.

If the operator takes the first option, there might be short downtimes during failovers.

If the operator takes the second option, the existing load balancers will not support upgraded Octavia API features, like UDP listeners. In this case, users must recreate their Services to use these features.

21.6.3.5. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.6.3.6. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

21.6.3.7. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.6.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

21.6.5. Downloading playbook dependencies

The Ansible playbooks that simplify the installation process on user-provisioned infrastructure require several Python modules. On the machine where you will run the installer, add the modules' repositories and then download them.



NOTE

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:
 - a. Register with Red Hat Subscription Manager:

```
$ sudo subscription-manager register # If not done already
```

- b. Pull the latest subscription data:

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

- c. Disable the current repositories:

```
$ sudo subscription-manager repos --disable=* # If not done already
```

- d. Add the required repositories:

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. Install the modules:

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
ansible-collections-openstack
```

3. Ensure that the **python** command points to **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

21.6.6. Downloading the installation playbooks

Download Ansible playbooks that you can use to install OpenShift Container Platform on your own Red Hat OpenStack Platform (RHOSP) infrastructure.

Prerequisites

- The curl command-line tool is available on your machine.

Procedure

- To download the playbooks to your working directory, run the following script from a command line:

```
$ xargs -n 1 curl -O <<< '
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/bootstrap.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/common.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/compute-nodes.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/control-
plane.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/inventory.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
4.12/upi/openstack/network.yaml
  https://raw.githubusercontent.com/openshift/installer/release-
```

```

4.12/upi/openstack/security-groups.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
bootstrap.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
compute-nodes.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
control-plane.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
load-balancers.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
network.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
security-groups.yaml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/openstack/down-
containers.yaml'

```

The playbooks are downloaded to your machine.



IMPORTANT

During the installation process, you can modify the playbooks to configure your deployment.

Retain all playbooks for the life of your cluster. You must have the playbooks to remove your OpenShift Container Platform cluster from RHOSP.



IMPORTANT

You must match any edits you make in the **bootstrap.yaml**, **compute-nodes.yaml**, **control-plane.yaml**, **network.yaml**, and **security-groups.yaml** files to the corresponding playbooks that are prefixed with **down-**. For example, edits to the **bootstrap.yaml** file must be reflected in the **down-bootstrap.yaml** file, too. If you do not edit both files, the supported cluster removal process will fail.

21.6.7. Obtaining the installation program

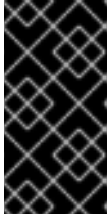
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

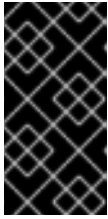
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

21.6.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Progress cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

21.6.9. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image

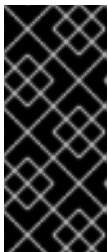
The OpenShift Container Platform installation program requires that a Red Hat Enterprise Linux CoreOS (RHCOS) image be present in the Red Hat OpenStack Platform (RHOSP) cluster. Retrieve the latest RHCOS image, then upload it using the RHOSP CLI.

Prerequisites

- The RHOSP CLI is installed.

Procedure

1. Log in to the Red Hat Customer Portal's [Product Downloads page](#).
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.12 for Red Hat Enterprise Linux (RHEL) 8.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the *Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)*.
4. Decompress the image.



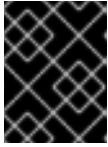
NOTE

You must decompress the RHOSP image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like **.gz** or **.tgz**. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. From the image that you downloaded, create an image that is named **rhcos** in your cluster by using the RHOSP CLI:

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-
${RHCOS_VERSION}-openstack.qcow2 rhcos
```



IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either [.raw](#) or [.qcow2](#) formats. If you use Ceph, you must use the [.raw](#) format.



WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

After you upload the image to RHOSP, it is usable in the installation process.

21.6.10. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- [Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries](#)

Procedure

1. Using the RHOSP CLI, verify the name and ID of the 'External' network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+-----+-----+-----+
| ID                | Name          | Router Type |
+-----+-----+-----+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+-----+-----+-----+
```

A network with an external router type appears in the network list. If at least one does not, see [Creating a default floating IP network](#) and [Creating a default provider network](#).



NOTE

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see [Neutron trunk port](#).

21.6.11. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

21.6.11.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API, cluster applications, and the bootstrap process.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"  
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"  
<external_network>
```

3. By using the Red Hat OpenStack Platform (RHOSP) CLI, create the bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>  
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```



NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your **/etc/hosts** file:

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

The cluster domain names in the **/etc/hosts** file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the **<application_floating_ip>**. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

5. Add the FIPs to the **inventory.yaml** file as the values of the following variables:

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

If you use these values, you must also enter an external network as the value of the **os_external_network** variable in the **inventory.yaml** file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

21.6.11.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the **inventory.yaml** file, do not define the following variables:

- **os_api_fip**
- **os_bootstrap_fip**

- **os_ingress_fip**

If you cannot provide an external network, you can also leave **os_external_network** blank. If you do not provide a value for **os_external_network**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. Later in the installation process, when you create network resources, you must configure external connectivity on your own.

If you run the installer with the **wait-for** command from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

21.6.12. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:
 - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.



IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in [a separate file](#) from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see [Config files](#) in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
```

```
dev-env:
  region_name: RegionOne
  auth:
    username: <username>
    password: <password>
    project_name: 'devonly'
    auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
 - a. Copy the certificate authority file to your machine.
 - b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
  ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
 - a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
 - b. The current directory
 - c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**
 - d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**
The installation program searches for **clouds.yaml** in that order.

21.6.13. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **openstack** as the platform to target.
 - iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.
 - iv. Specify the floating IP address to use for external access to the OpenShift API.
 - v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.
 - vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.
 - vii. Enter a name for your cluster. The name must be 14 or fewer characters long.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

You now have the file **install-config.yaml** in the directory that you specified.

21.6.14. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

21.6.14.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 21.20. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev . The string must be 14 characters or fewer long.
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

21.6.14.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 21.21. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


21.6.14.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 21.22. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}

Parameter	Description	Values
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div data-bbox="485 1473 592 1792" style="display: inline-block; vertical-align: middle;">  </div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 50px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

21.6.14.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 21.23. Additional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.rootVolume.size	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .

Parameter	Description	Values
compute.platform.openstack.rootVolume.type	For compute machines, the root volume's type.	String, for example performance .
controlPlane.platform.openstack.rootVolume.size	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
controlPlane.platform.openstack.rootVolume.type	For control plane machines, the root volume's type.	String, for example performance .
platform.openstack.cloud	The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.	String, for example MyCloud .
platform.openstack.externalNetwork	The RHOSP external network name to be used for installation.	String, for example external .
platform.openstack.computeFlavor	<p>The RHOSP flavor to use for control plane and compute machines.</p> <p>This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the type key in the platform.openstack.defaultMachinePlatform property. You can also set a flavor value for each machine pool individually.</p>	String, for example m1.xlarge .

21.6.14.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 21.24. Optional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.additionalNetworkIDs	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .
compute.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .
compute.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
compute.platform.openstack.rootVolume.zones	For compute machines, the availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .

Parameter	Description	Values
compute.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	<p>A server group policy to apply to the machine pool. For example, soft-affinity.</p>
controlPlane.platform.openstack.additionalNetworkIDs	<p>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</p> <p>Additional networks that are attached to a control plane machine are also attached to the bootstrap node.</p>	<p>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</p>
controlPlane.platform.openstack.additionalSecurityGroupIDs	<p>Additional security groups that are associated with control plane machines.</p>	<p>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</p>

Parameter	Description	Values
controlPlane.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
controlPlane.platform.openstack.rootVolume.zones	For control plane machines, the availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .
controlPlane.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations, and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .

Parameter	Description	Values
platform.openstack.clusterOSImage	<p>The location from which the installation program downloads the RHCOS image.</p> <p>You must set this parameter to perform an installation in a restricted network.</p>	<p>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.</p> <p>For example, http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d. The value can also be the name of an existing Glance image, for example my-rhcos.</p>
platform.openstack.clusterOSImageProperties	<p>Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if platform.openstack.clusterOSImage is set to an existing Glance image.</p> <p>You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the hw_scsi_model property value to virtio-scsi and the hw_disk_bus value to scsi.</p> <p>You can also use this property to enable the QEMU guest agent by including the hw_qemu_guest_agent property with a value of yes.</p>	<p>A list of key-value string pairs. For example, ["hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"].</p>
platform.openstack.defaultMachinePlatform	<p>The default machine pool platform configuration.</p>	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</p>	<p>An IP address, for example 128.0.0.1.</p>

Parameter	Description	Values
platform.openstack.apiFloatingIP	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.	An IP address, for example 128.0.0.1 .
platform.openstack.externalDNS	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, ["8.8.8.8", "192.168.1.12"] .
platform.openstack.machinesSubnet	<p>The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</p> <p>The first item in networking.machineNetwork must match the value of machinesSubnet.</p> <p>If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</p>	A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

21.6.14.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

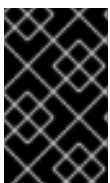
Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.
- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.



NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.



IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

21.6.14.7. Sample customized **install-config.yaml** file for RHOSP with Kuryr

To deploy with Kuryr SDN instead of the default OVN-Kubernetes network plugin, you must modify the **install-config.yaml** file to include **Kuryr** as the desired **networking.networkType**. This sample **install-config.yaml** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.



IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
```

```
- cidr: 10.0.0.0/16
serviceNetwork:
- 172.30.0.0/16 1
networkType: Kuryr 2
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
    trunkSupport: true 3
    octaviaSupport: true 4
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

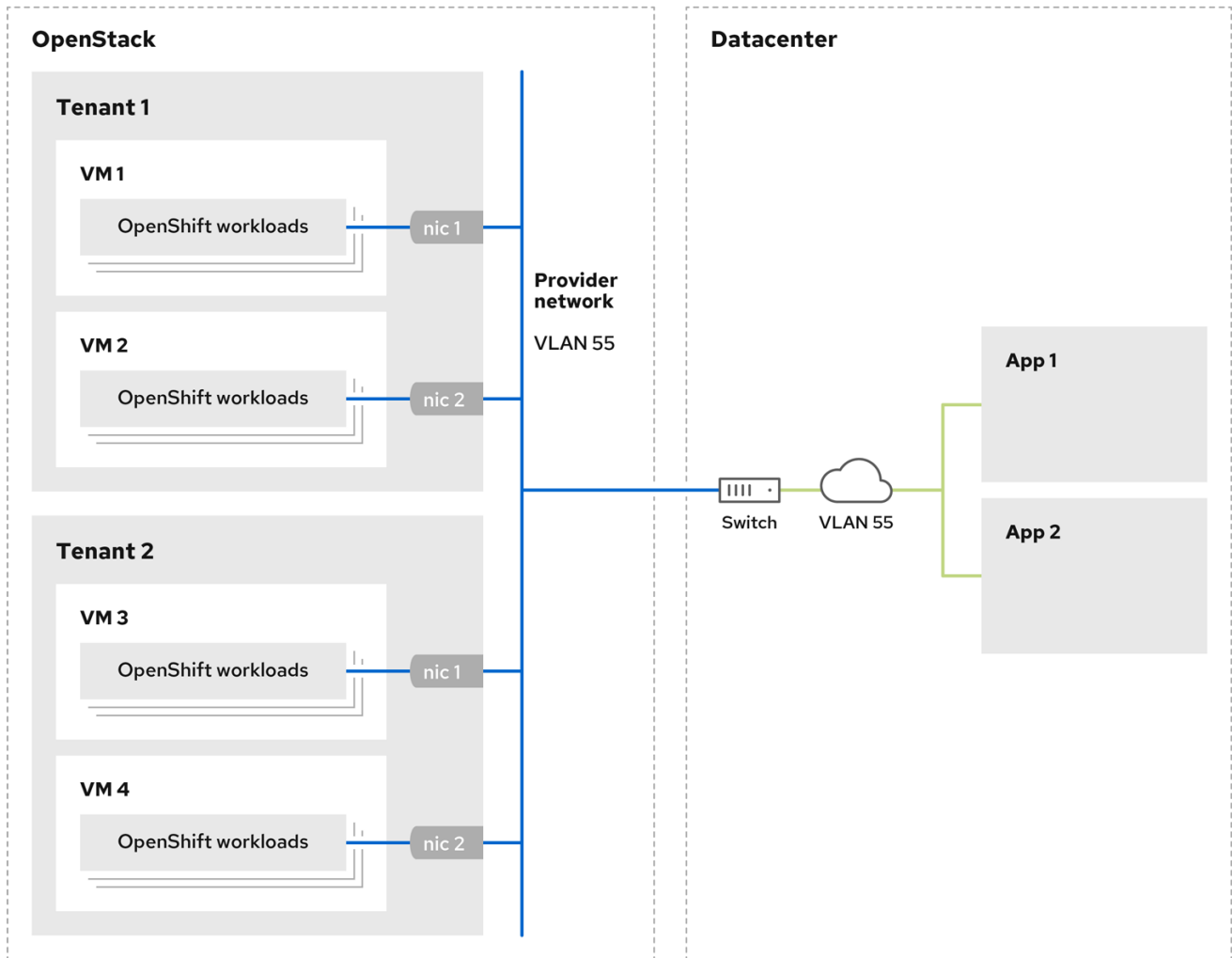
- 1 The Amphora Octavia driver creates two ports per load balancer. As a result, the service subnet that the installer creates is twice the size of the CIDR that is specified as the value of the **serviceNetwork** property. The larger range is required to prevent IP address conflicts.
- 2 The cluster network plugin to install. The supported values are **Kuryr**, **OVNKubernetes**, and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 3 4 Both **trunkSupport** and **octaviaSupport** are automatically discovered by the installer, so there is no need to set them. But if your environment does not meet both requirements, Kuryr SDN will not properly work. Trunks are needed to connect the pods to the RHOSP network and Octavia is required to create the OpenShift Container Platform services.

21.6.14.8. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

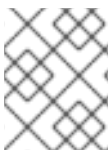
In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



170_OpenShift_0621

OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).



NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in [the RHOSP documentation](#).

21.6.14.8.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The [RHOSP networking service \(Neutron\) is enabled](#) and accessible through the RHOSP networking API.
- The RHOSP networking service has the [port security and allowed address pairs extensions enabled](#).

- The provider network can be shared with other tenants.

TIP

Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

TIP

To create a network for a project that is named "openshift," enter the following command

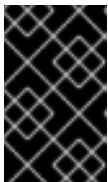
```
$ openstack network create --project openshift
```

To create a subnet for a project that is named "openshift," enter the following command

```
$ openstack subnet create --project openshift
```

To learn more about creating networks on RHOSP, read [the provider networks documentation](#).

If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.



IMPORTANT

Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.

Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

```
$ openstack subnet create --dhcp --host-route  
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- Optional: To secure the network, create [role-based access control \(RBAC\)](#) rules that limit network access to a single project.

21.6.14.8.2. Deploying a cluster that has a primary interface on a provider network

You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.



IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: ①
      - 192.0.2.13
    ingressVIPs: ②
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
  networking:
    machineNetwork:
      - cidr: 192.0.2.0/24
```

- ① ② In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.



WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

TIP

You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see [Understanding multiple networks](#).

21.6.14.9. Kuryr ports pools

A Kuryr ports pool maintains a number of ports on standby for pod creation.

Keeping ports on standby minimizes pod creation time. Without ports pools, Kuryr must explicitly request port creation or deletion whenever a pod is created or deleted.

The Neutron ports that Kuryr uses are created in subnets that are tied to namespaces. These pod ports are also added as subports to the primary port of OpenShift Container Platform cluster nodes.

Because Kuryr keeps each namespace in a separate subnet, a separate ports pool is maintained for each namespace-worker pair.

Prior to installing a cluster, you can set the following parameters in the **cluster-network-03-config.yml** manifest file to configure ports pool behavior:

- The **enablePortPoolsPrepopulation** parameter controls pool prepopulation, which forces Kuryr to add Neutron ports to the pools when the first pod that is configured to use the dedicated network for pods is created in a namespace. The default value is **false**.
- The **poolMinPorts** parameter is the minimum number of free ports that are kept in the pool. The default value is **1**.
- The **poolMaxPorts** parameter is the maximum number of free ports that are kept in the pool. A value of **0** disables that upper bound. This is the default setting.
If your OpenStack port quota is low, or you have a limited number of IP addresses on the pod network, consider setting this option to ensure that unneeded ports are deleted.
- The **poolBatchPorts** parameter defines the maximum number of Neutron ports that can be created at once. The default value is **3**.

21.6.14.10. Adjusting Kuryr ports pools during installation

During installation, you can configure how Kuryr manages Red Hat OpenStack Platform (RHOSP) Neutron ports to control the speed and efficiency of pod creation.

Prerequisites

- Create and modify the **install-config.yaml** file.

Procedure

1. From a command line, create the manifest files:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the name of the directory that contains the **install-config.yaml** file for your cluster.
2. Create a file that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

- 1 For **<installation_directory>**, specify the directory name that contains the **manifests/** directory for your cluster.

After creating the file, several network configuration files are in the **manifests/** directory, as shown:

```
$ ls <installation_directory>/manifests/cluster-network-*
```

Example output

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. Open the **cluster-network-03-config.yml** file in an editor, and enter a custom resource (CR) that describes the Cluster Network Operator configuration that you want:

```
$ oc edit networks.operator.openshift.io cluster
```

4. Edit the settings to meet your requirements. The following file is provided as an example:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
      enablePortPoolsPrepopulation: false 1
      poolMinPorts: 1 2
      poolBatchPorts: 3 3
      poolMaxPorts: 5 4
      openstackServiceNetwork: 172.30.0.0/15 5
```

- 1 Set **enablePortPoolsPrepopulation** to **true** to make Kuryr create new Neutron ports when the first pod on the network for pods is created in a namespace. This setting raises the Neutron ports quota but can reduce the time that is required to spawn pods. The default value is **false**.

- 2 Kuryr creates new ports for a pool if the number of free ports in that pool is lower than the value of **poolMinPorts**. The default value is **1**.
- 3 **poolBatchPorts** controls the number of new ports that are created if the number of free ports is lower than the value of **poolMinPorts**. The default value is **3**.
- 4 If the number of free ports in a pool is higher than the value of **poolMaxPorts**, Kuryr deletes them until the number matches that value. Setting this value to **0** disables this upper bound, preventing pools from shrinking. The default value is **0**.
- 5 The **openStackServiceNetwork** parameter defines the CIDR range of the network from which IP addresses are allocated to RHOSP Octavia's LoadBalancers.

If this parameter is used with the Amphora driver, Octavia takes two IP addresses from this network for each load balancer: one for OpenShift and the other for VRRP connections. Because these IP addresses are managed by OpenShift Container Platform and Neutron respectively, they must come from different pools. Therefore, the value of **openStackServiceNetwork** must be at least twice the size of the value of **serviceNetwork**, and the value of **serviceNetwork** must overlap entirely with the range that is defined by **openStackServiceNetwork**.

The CNO verifies that VRRP IP addresses that are taken from the range that is defined by this parameter do not overlap with the range that is defined by the **serviceNetwork** parameter.

If this parameter is not set, the CNO uses an expanded value of **serviceNetwork** that is determined by decrementing the prefix size by 1.

5. Save the **cluster-network-03-config.yml** file, and exit the text editor.
6. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory while creating the cluster.

21.6.14.11. Setting a custom subnet for machines

The IP range that the installation program uses by default might not match the Neutron subnet that you create when you install OpenShift Container Platform. If necessary, update the CIDR value for new machines by editing the installation configuration file.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.

Procedure

1. On a command line, browse to the directory that contains **install-config.yaml**.
2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:
 - To set the value by using a script, run:

```
$ python -c '  
import yaml;  
path = "install-config.yaml";
```



```
data = yaml.safe_load(open(path));
data["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; 1
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- 1** Insert a value that matches your intended Neutron subnet, e.g. **192.0.2.0/24**.

- To set the value manually, open the file and set the value of **networking.machineCIDR** to something that matches your intended Neutron subnet.

21.6.14.12. Emptying compute machine pools

To proceed with an installation that uses your own infrastructure, set the number of compute machines in the installation configuration file to zero. Later, you create these machines manually.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.

Procedure

1. On a command line, browse to the directory that contains **install-config.yaml**.
2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:
 - To set the value by using a script, run:

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- To set the value manually, open the file and set the value of **compute.<first entry>.replicas** to **0**.

21.6.14.13. Modifying the network type

By default, the installation program selects the **OpenShiftSDN** network type. To use Kuryr instead, change the value in the installation configuration file that the program generated.

Prerequisites

- You have the file **install-config.yaml** that was generated by the OpenShift Container Platform installation program

Procedure

1. In a command prompt, browse to the directory that contains **install-config.yaml**.
2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:

- To set the value by using a script, run:

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["networking"]["networkType"] = "Kuryr";
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- To set the value manually, open the file and set **networking.networkType** to **"Kuryr"**.

21.6.15. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

- Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
- Check that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - Save and exit the file.
 - To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

- Export the metadata file's **infraID** key as an environment variable:

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

TIP

Extract the **infraID** key from **metadata.json** and use it as a prefix for all of the RHOSP resources that you create. By doing so, you avoid name conflicts when making multiple deployments in the same project.

21.6.16. Preparing the bootstrap Ignition files

The OpenShift Container Platform installation process relies on bootstrap machines that are created from a bootstrap Ignition configuration file.

Edit the file and upload it. Then, create a secondary bootstrap Ignition configuration file that Red Hat OpenStack Platform (RHOSP) uses to download the primary file.

Prerequisites

- You have the bootstrap Ignition file that the installer program generates, **bootstrap.ign**.
- The infrastructure ID from the installer's metadata file is set as an environment variable (**\$INFRA_ID**).
 - If the variable is not set, see **Creating the Kubernetes manifest and Ignition config files**
- You have an HTTP(S)-accessible way to store the bootstrap Ignition file.
 - The documented procedure uses the RHOSP image service (Glance), but you can also use the RHOSP storage service (Swift), Amazon S3, an internal HTTP server, or an ad hoc Nova server.

Procedure

1. Run the following Python script. The script modifies the bootstrap Ignition file to set the hostname and, if available, CA certificate file when it runs:

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', "")
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

files.append(
{
    'path': '/opt/openshift/tls/cloud-ca-cert.pem',
    'mode': 420,
```

```

        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

    ignition['storage']['files'] = files;

    with open('bootstrap.ign', 'w') as f:
        json.dump(ignition, f)

```

- Using the RHOSP CLI, create an image that uses the bootstrap ignition file:

```

$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>

```

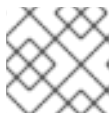
- Get the image's details:

```

$ openstack image show <image_name>

```

Make a note of the **file** value; it follows the pattern **v2/images/<image_ID>/file**.



NOTE

Verify that the image you created is active.

- Retrieve the image service's public address:

```

$ openstack catalog show image

```

- Combine the public address with the image **file** value and save the result as the storage location. The location follows the pattern **<image_service_public_URL>/v2/images/<image_ID>/file**.

- Generate an auth token and save the token ID:

```

$ openstack token issue -c id -f value

```

- Insert the following content into a file called **\$INFRA_ID-bootstrap-ignition.json** and edit the placeholders to match your own values:

```

{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {

```

```

    "certificateAuthorities": [{
      "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
    }]
  }
},
"version": "3.2.0"
}
}

```

- 1 Replace the value of **ignition.config.merge.source** with the bootstrap Ignition file storage URL.
- 2 Set **name** in **httpHeaders** to **"X-Auth-Token"**.
- 3 Set **value** in **httpHeaders** to your token's ID.
- 4 If the bootstrap Ignition file server uses a self-signed certificate, include the base64-encoded certificate.

8. Save the secondary Ignition config file.

The bootstrap Ignition data will be passed to RHOSP during installation.

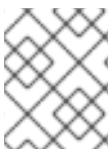


WARNING

The bootstrap Ignition file contains sensitive information, like **clouds.yaml** credentials. Ensure that you store it in a secure place, and delete it after you complete the installation process.

21.6.17. Creating control plane Ignition config files on RHOSP

Installing OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) on your own infrastructure requires control plane Ignition config files. You must create multiple config files.



NOTE

As with the bootstrap Ignition configuration, you must explicitly define a hostname for each control plane machine.

Prerequisites

- The infrastructure ID from the installation program's metadata file is set as an environment variable (**\$INFRA_ID**).
 - If the variable is not set, see "Creating the Kubernetes manifest and Ignition config files".

Procedure

- On a command line, run the following Python script:

■

```

$ for index in $(seq 0 2); do
  MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
  python -c "import base64, json, sys;
  ignition = json.load(sys.stdin);
  storage = ignition.get('storage', {});
  files = storage.get('files', []);
  files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
'filesystem': 'root'});
  storage['files'] = files;
  ignition['storage'] = storage;
  json.dump(ignition, sys.stdout) <master.ign >"$INFRA_ID-master-$index-ignition.json"
done

```

You now have three control plane Ignition files: **<INFRA_ID>-master-0-ignition.json**, **<INFRA_ID>-master-1-ignition.json**, and **<INFRA_ID>-master-2-ignition.json**.

21.6.18. Creating network resources on RHOSP

Create the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires. To save time, run supplied Ansible playbooks that generate security groups, networks, subnets, routers, and ports.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".

Procedure

1. Optional: Add an external network value to the **inventory.yaml** playbook:

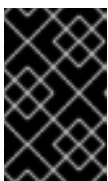
Example external network value in the **inventory.yaml** Ansible playbook

```

...
# The public network providing connectivity to the cluster. If not
# provided, the cluster external connectivity must be provided in another
# way.

# Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
os_external_network: 'external'
...

```



IMPORTANT

If you did not provide a value for **os_external_network** in the **inventory.yaml** file, you must ensure that VMs can access Glance and an external connection yourself.

- Optional: Add external network and floating IP (FIP) address values to the **inventory.yaml** playbook:

Example FIP values in the **inventory.yaml** Ansible playbook

```
...
# OpenShift API floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the Control Plane to
# serve the OpenShift API.
os_api_fip: '203.0.113.23'

# OpenShift Ingress floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the worker nodes to serve
# the applications.
os_ingress_fip: '203.0.113.19'

# If this value is non-empty, the corresponding floating IP will be
# attached to the bootstrap machine. This is needed for collecting logs
# in case of install failure.
os_bootstrap_fip: '203.0.113.20'
```



IMPORTANT

If you do not define values for **os_api_fip** and **os_ingress_fip**, you must perform postinstallation network configuration.

If you do not define a value for **os_bootstrap_fip**, the installer cannot download debugging information from failed installations.

See "Enabling access to the environment" for more information.

- On a command line, create security groups by running the **security-groups.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

- On a command line, create a network, subnet, and router by running the **network.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml network.yaml
```

- Optional: If you want to control the default resolvers that Nova servers use, run the RHOSP CLI command:

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

21.6.19. Creating the bootstrap machine on RHOSP

Create a bootstrap machine and give it the network access it needs to run on Red Hat OpenStack Platform (RHOSP). Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **bootstrap.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **bootstrap.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. After the bootstrap server is active, view the logs to verify that the Ignition files were received:

```
$ openstack console log show "$INFRA_ID-bootstrap"
```

21.6.20. Creating the control plane machines on RHOSP

Create three control plane machines by using the Ignition config files that you generated. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The infrastructure ID from the installation program's metadata file is set as an environment variable (**\$INFRA_ID**).
- The **inventory.yaml**, **common.yaml**, and **control-plane.yaml** Ansible playbooks are in a common directory.
- You have the three Ignition files that were created in "Creating control plane Ignition config files".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. If the control plane Ignition config files aren't already in your working directory, copy them into it.
3. On a command line, run the **control-plane.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. Run the following command to monitor the bootstrapping process:

```
■
```

```
$ openshift-install wait-for bootstrap-complete
```

You will see messages that confirm that the control plane machines are running and have joined the cluster:

```
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

21.6.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

21.6.22. Deleting bootstrap resources from RHOSP

Delete the bootstrap resources that you no longer need.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **down-bootstrap.yaml** Ansible playbooks are in a common directory.

- The control plane machines are running.
 - If you do not know the status of the machines, see "Verifying cluster status".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **down-bootstrap.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

The bootstrap port, server, and floating IP address are deleted.



WARNING

If you did not disable the bootstrap Ignition file URL earlier, do so now.

21.6.23. Creating compute machines on RHOSP

After standing up the control plane, create compute machines. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **compute-nodes.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.
- The control plane is active.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

Next steps

- Approve the certificate signing requests for the machines.

21.6.24. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   63m   v1.25.0
master-1  Ready   master   63m   v1.25.0
master-2  Ready   master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

21.6.25. Verifying a successful installation

Verify that the OpenShift Container Platform installation is complete.

Prerequisites

- You have the installation program (**openshift-install**)

Procedure

- On a command line, enter:

```
$ openshift-install --log-level debug wait-for install-complete
```

The program outputs the console URL, as well as the administrator's login information.

21.6.26. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

21.6.27. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .
- If you need to enable external access to node ports, [configure ingress cluster traffic by using a node port](#).
- If you did not configure RHOSP to accept application traffic over floating IP addresses, [configure RHOSP access with floating IP addresses](#) .

21.7. INSTALLING A CLUSTER ON OPENSTACK IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.12, you can install a cluster on Red Hat OpenStack Platform (RHOSP) in a restricted network by creating an internal mirror of the installation release content.

21.7.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You verified that OpenShift Container Platform 4.12 is compatible with your RHOSP version by using the [Supported platforms for OpenShift clusters](#) section. You can also compare platform support across different versions by viewing the [OpenShift Container Platform on RHOSP support matrix](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see [Recommended practices for scaling the cluster](#).
- You have the metadata service enabled in RHOSP.

21.7.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

21.7.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

21.7.3. Resource guidelines for installing OpenShift Container Platform on RHOSP

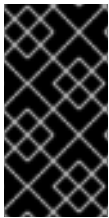
To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 21.25. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

Resource	Value
Floating IP addresses	3
Ports	15
Routers	1
Subnets	1
RAM	88 GB
vCPUs	22

Resource	Value
Volume storage	275 GB
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

21.7.3.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.7.3.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

21.7.3.3. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

21.7.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

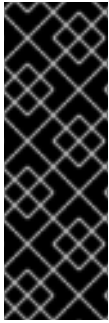


IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

21.7.5. Enabling Swift on RHOSP

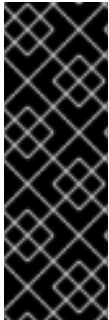
Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.



IMPORTANT

If [the Red Hat OpenStack Platform \(RHOSP\) object storage service](#), commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.



IMPORTANT

RHOSP 17 sets the **rgw_max_attr_size** parameter of Ceph RGW to 256 characters. This setting causes issues with uploading container images to the OpenShift Container Platform registry. You must set the value of **rgw_max_attr_size** to at least 1024 characters.

Before installation, check if your RHOSP deployment is affected by this problem. If it is, reconfigure Ceph RGW.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On [Ceph RGW](#), the **account in url** option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

```
$ openstack role add --user <user> --project <project> swiftoperator
```

Your RHOSP deployment can now use Swift for the image registry.

21.7.6. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:
 - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.

**IMPORTANT**

Remember to add a password to the **auth** field. You can also keep secrets in a [separate file](#) from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see [Config files](#) in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
 - a. Copy the certificate authority file to your machine.
 - b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
 - a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
 - b. The current directory
 - c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**

- d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`
The installation program searches for `clouds.yaml` in that order.

21.7.7. Setting OpenStack Cloud Controller Manager options

Optionally, you can edit the OpenStack Cloud Controller Manager (CCM) configuration for your cluster. This configuration controls how OpenShift Container Platform interacts with Red Hat OpenStack Platform (RHOSP).

For a complete list of configuration parameters, see the "OpenStack Cloud Controller Manager reference guide" page in the "Installing on OpenStack" documentation.

Procedure

1. If you have not already generated manifest files for your cluster, generate them by running the following command:

```
$ openshift-install --dir <destination_directory> create manifests
```

2. In a text editor, open the cloud-provider configuration manifest file. For example:

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. Modify the options according to the CCM reference guide.
Configuring Octavia for load balancing is a common case for clusters that do not use Kuryr. For example:

```
#...
[LoadBalancer]
use-octavia=true 1
lb-provider = "amphora" 2
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 3
create-monitor = True 4
monitor-delay = 10s 5
monitor-timeout = 10s 6
monitor-max-retries = 1 7
#...
```

- 1 This property enables Octavia integration.
- 2 This property sets the Octavia provider that your load balancer uses. It accepts "ovn" or "amphora" as values. If you choose to use OVN, you must also set `lb-method` to `SOURCE_IP_PORT`.
- 3 This property is required if you want to use multiple external networks with your cluster. The cloud provider creates floating IP addresses on the network that is specified here.
- 4 This property controls whether the cloud provider creates health monitors for Octavia load balancers. Set the value to `True` to create health monitors. As of RHOSP 16.2, this feature is only available for the Amphora provider.
- 5 This property sets the frequency with which endpoints are monitored. The value must be in the `time.ParseDuration()` format. This property is required if the value of the `create-monitor` property is `True`.

- 6 This property sets the time that monitoring requests are open before timing out. The value must be in the **time.ParseDuration()** format. This property is required if the value of the
- 7 This property defines how many successful monitoring requests are required before a load balancer is marked as online. The value must be an integer. This property is required if the value of the **create-monitor** property is **True**.



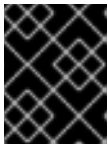
IMPORTANT

Prior to saving your changes, verify that the file is structured correctly. Clusters might fail if properties are not placed in the appropriate section.



IMPORTANT

You must set the value of the **create-monitor** property to **True** if you use services that have the value of the **.spec.externalTrafficPolicy** property set to **Local**. The OVN Octavia provider in RHOSP 16.2 does not support health monitors. Therefore, services that have **ETP** parameter values set to **Local** might not respond when the **lb-provider** value is set to **"ovn"**.



IMPORTANT

For installations that use Kuryr, Kuryr handles relevant services. There is no need to configure Octavia load balancing in the cloud provider.

4. Save the changes to the file and proceed with installation.

TIP

You can update your cloud provider configuration after you run the installer. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

After you save your changes, your cluster will take some time to reconfigure itself. The process is complete if none of your nodes have a **SchedulingDisabled** status.

21.7.8. Creating the RHCOS image for restricted network installations

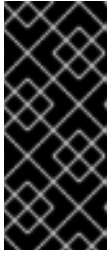
Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network Red Hat OpenStack Platform (RHOSP) environment.

Prerequisites

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

Procedure

1. Log in to the Red Hat Customer Portal's [Product Downloads page](#).
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.12 for RHEL 8.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)** image.
4. Decompress the image.



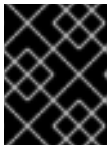
NOTE

You must decompress the image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like **.gz** or **.tgz**. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. Upload the image that you decompressed to a location that is accessible from the bastion server, like Glance. For example:

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --disk-format qcow2 rhcos- $\{RHCOS\_VERSION\}$ 
```



IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either **.raw** or **.qcow2** formats. If you use Ceph, you must use the **.raw** format.



WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

21.7.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **openstack** as the platform to target.
- iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.
- iv. Specify the floating IP address to use for external access to the OpenShift API.
- v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

- vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.
 - vii. Enter a name for your cluster. The name must be 14 or fewer characters long.
 - viii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. In the **install-config.yaml** file, set the value of **platform.openstack.clusterOSImage** to the image location or name. For example:

```
platform:
  openstack:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
openstack.x86_64.qcow2.gz?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000":{"auth": "<credentials>","email":
"you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----/
  -----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

4. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

21.7.9.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.



NOTE

Kuryr installations default to HTTP proxies.

Prerequisites

- For Kuryr installations on restricted networks that use the **Proxy** object, the proxy must be able to reply to the router that the cluster uses. To add a static route for the proxy configuration, from a command line as the root user, enter:

```
$ ip route add <cluster_network_cidr> via <installer_subnet_gateway>
```

- The restricted subnet must have a gateway that is defined and available to be linked to the **Router** resource that Kuryr creates.
- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

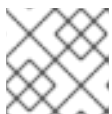
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

21.7.9.2. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

21.7.9.2.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 21.26. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> , <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object

Parameter	Description	Values
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev . The string must be 14 characters or fewer long.
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

21.7.9.2.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

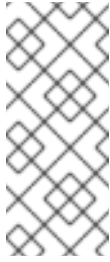


NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 21.27. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>

Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


21.7.9.2.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 21.28. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 20px; height: 100px; margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}

Parameter	Description	Values
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, #ccc 2px, #ccc 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, #ccc 2px, #ccc 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").
fips	Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.	false or true

Parameter	Description	Values
	<p data-bbox="671 109 863 141">IMPORTANT</p> <p data-bbox="671 181 932 909">To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="671 965 762 996">NOTE</p> <p data-bbox="671 1032 922 1151">If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings

Parameter	Description	Values
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

21.7.9.2.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 21.29. Additional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.rootVolume.size	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
compute.platform.openstack.rootVolume.type	For compute machines, the root volume's type.	String, for example performance .

Parameter	Description	Values
controlPlane.platform.openstack.rootVolume.size	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example 30 .
controlPlane.platform.openstack.rootVolume.type	For control plane machines, the root volume's type.	String, for example performance .
platform.openstack.cloud	The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.	String, for example MyCloud .
platform.openstack.externalNetwork	The RHOSP external network name to be used for installation.	String, for example external .
platform.openstack.computeFlavor	<p>The RHOSP flavor to use for control plane and compute machines.</p> <p>This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the type key in the platform.openstack.defaultMachinePlatform property. You can also set a flavor value for each machine pool individually.</p>	String, for example m1.xlarge .

21.7.9.2.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 21.30. Optional RHOSP parameters

Parameter	Description	Values
compute.platform.openstack.additionalNetworks	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

Parameter	Description	Values
compute.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .
compute.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
compute.platform.openstack.rootVolume.zones	For compute machines, the availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .

Parameter	Description	Values
compute.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .
controlPlane.platform.openstack.additionalNetworkIDs	<p>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</p> <p>Additional networks that are attached to a control plane machine are also attached to the bootstrap node.</p>	A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .
controlPlane.platform.openstack.additionalSecurityGroupIDs	Additional security groups that are associated with control plane machines.	A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 .

Parameter	Description	Values
controlPlane.platform.openstack.zones	<p>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.</p> <p>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</p>	A list of strings. For example, ["zone-1", "zone-2"] .
controlPlane.platform.openstack.rootVolume.zones	For control plane machines, the availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone.	A list of strings, for example ["zone-1", "zone-2"] .
controlPlane.platform.openstack.serverGroupPolicy	<p>Server group policy to apply to the group that will contain the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include anti-affinity, soft-affinity, and soft-anti-affinity. The default value is soft-anti-affinity.</p> <p>An affinity policy prevents migrations, and therefore affects RHOSP upgrades. The affinity policy is not supported.</p> <p>If you use a strict anti-affinity policy, an additional RHOSP host is required during instance migration.</p>	A server group policy to apply to the machine pool. For example, soft-affinity .

Parameter	Description	Values
platform.openstack.clusterOSImage	<p>The location from which the installation program downloads the RHCOS image.</p> <p>You must set this parameter to perform an installation in a restricted network.</p>	<p>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.</p> <p>For example, http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d. The value can also be the name of an existing Glance image, for example my-rhcos.</p>
platform.openstack.clusterOSImageProperties	<p>Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if platform.openstack.clusterOSImage is set to an existing Glance image.</p> <p>You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the hw_scsi_model property value to virtio-scsi and the hw_disk_bus value to scsi.</p> <p>You can also use this property to enable the QEMU guest agent by including the hw_qemu_guest_agent property with a value of yes.</p>	<p>A list of key-value string pairs. For example, ["hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"].</p>
platform.openstack.defaultMachinePlatform	<p>The default machine pool platform configuration.</p>	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</p>	<p>An IP address, for example 128.0.0.1.</p>

Parameter	Description	Values
platform.openstack.apiFloatingIP	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.	An IP address, for example 128.0.0.1 .
platform.openstack.externalDNS	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, ["8.8.8.8", "192.168.1.12"] .
platform.openstack.machinesSubnet	<p>The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</p> <p>The first item in networking.machineNetwork must match the value of machinesSubnet.</p> <p>If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</p>	A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf .

21.7.9.3. Sample customized `install-config.yaml` file for restricted OpenStack installations

This sample **`install-config.yaml`** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.



IMPORTANT

This sample file is provided for reference only. You must obtain your **`install-config.yaml`** file by using the installation program.

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
```

```

replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  serviceNetwork:
    - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle: |

-----BEGIN CERTIFICATE-----

////////////////////////////////////

-----END CERTIFICATE-----

imageContentSources:
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev

```

21.7.10. Generating a key pair for cluster node SSH access

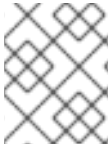
During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

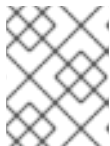
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

21.7.11. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

21.7.11.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```

NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your `/etc/hosts` file:

- `<api_floating_ip> api.<cluster_name>.<base_domain>`
- `<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>`
- `<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>`
- `application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>`

The cluster domain names in the `/etc/hosts` file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the `<application_floating_ip>`. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

4. Add the FIPs to the `install-config.yaml` file as the values of the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.apiFloatingIP`

If you use these values, you must also enter an external network as the value of the `platform.openstack.externalNetwork` parameter in the `install-config.yaml` file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

21.7.11.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `install-config.yaml` file, do not define the following parameters:

- `platform.openstack.ingressFloatingIP`

- **platform.openstack.apiFloatingIP**

If you cannot provide an external network, you can also leave **platform.openstack.externalNetwork** blank. If you do not provide a value for **platform.openstack.externalNetwork**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

21.7.12. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

21.7.13. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:


```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

- View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

- View your cluster's version:

```
$ oc get clusterversion
```

- View your Operators' status:

```
$ oc get clusteroperator
```

- View all running pods in the cluster:

```
$ oc get pods -A
```

21.7.14. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

- Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

21.7.15. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
-p [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

21.7.16. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

21.7.17. Next steps

- [Customize your cluster](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, see [Registering your disconnected cluster](#)

- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#).
- If you did not configure RHOSP to accept application traffic over floating IP addresses, [configure RHOSP access with floating IP addresses](#).

21.8. OPENSTACK CLOUD CONTROLLER MANAGER REFERENCE GUIDE

21.8.1. The OpenStack Cloud Controller Manager

In OpenShift Container Platform 4.12, clusters that run on Red Hat OpenStack Platform (RHOSP) are switched from the legacy OpenStack cloud provider to the external OpenStack Cloud Controller Manager (CCM). This change follows the move in Kubernetes from in-tree, legacy cloud providers to external cloud providers that are implemented by using the [Cloud Controller Manager](#).

To preserve user-defined configurations for the legacy cloud provider, existing configurations are mapped to new ones as part of the migration process. It searches for a configuration called **cloud-provider-config** in the **openshift-config** namespace.



NOTE

The config map name **cloud-provider-config** is not statically configured. It is derived from the **spec.cloudConfig.name** value in the **infrastructure/cluster** CRD.

Found configurations are synchronized to the **cloud-conf** config map in the **openshift-cloud-controller-manager** namespace.

As part of this synchronization, the OpenStack CCM Operator alters the new config map such that its properties are compatible with the external cloud provider. The file is changed in the following ways:

- The **[Global] secret-name**, **[Global] secret-namespace**, and **[Global] kubeconfig-path** options are removed. They do not apply to the external cloud provider.
- The **[Global] use-clouds**, **[Global] clouds-file**, and **[Global] cloud** options are added.
- The entire **[BlockStorage]** section is removed. External cloud providers no longer perform storage operations. Block storage configuration is managed by the Cinder CSI driver.

Additionally, the CCM Operator enforces a number of default options. Values for these options are always overridden as follows:

```
[Global]
use-clouds = true
clouds-file = /etc/openstack/secret/clouds.yaml
cloud = openstack
...

[LoadBalancer]
use-octavia = true
enabled = true 1
```

1 If the network is configured to use Kuryr, this value is **false**.

The **clouds-value** value, `/etc/openstack/secret/clouds.yaml`, is mapped to the **openstack-cloud-credentials** config in the **openshift-cloud-controller-manager** namespace. You can modify the RHOSP cloud in this file as you do any other **clouds.yaml** file.

21.8.2. The OpenStack Cloud Controller Manager (CCM) config map

An OpenStack CCM config map defines how your cluster interacts with your RHOSP cloud. By default, this configuration is stored under the **cloud.conf** key in the **cloud.conf** config map in the **openshift-cloud-controller-manager** namespace.



IMPORTANT

The **cloud.conf** config map is generated from the **cloud-provider-config** config map in the **openshift-config** namespace.

To change the settings that are described by the **cloud.conf** config map, modify the **cloud-provider-config** config map.

As part of this synchronization, the CCM Operator overrides some options. For more information, see "The RHOSP Cloud Controller Manager".

For example:

An example **cloud.conf** config map

```
apiVersion: v1
data:
  cloud.conf: |
    [Global] 1
    secret-name = openstack-credentials
    secret-namespace = kube-system
    region = regionOne
    [LoadBalancer]
    use-octavia = True
kind: ConfigMap
metadata:
  creationTimestamp: "2022-12-20T17:01:08Z"
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
  resourceVersion: "2519"
  uid: cbbeedaf-41ed-41c2-9f37-4885732d3677
```

1 Set global options by using a **clouds.yaml** file rather than modifying the config map.

The following options are present in the config map. Except when indicated otherwise, they are mandatory for clusters that run on RHOSP.

21.8.2.1. Load balancer options

CCM supports several load balancer options for deployments that use Octavia.

**NOTE**

Neutron-LBaaS support is deprecated.

Option	Description
enabled	Whether or not to enable the LoadBalancer type of services integration. The default value is true .
floating-network-id	Optional. The external network used to create floating IP addresses for load balancer virtual IP addresses (VIPs). If there are multiple external networks in the cloud, this option must be set or the user must specify loadbalancer.openstack.org/floating-network-id in the service annotation.
floating-subnet-id	Optional. The external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation loadbalancer.openstack.org/floating-subnet-id .
floating-subnet	Optional. A name pattern (glob or regular expression if starting with ~) for the external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation loadbalancer.openstack.org/floating-subnet . If multiple subnets match the pattern, the first one with available IP addresses is used.
floating-subnet-tags	Optional. Tags for the external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation loadbalancer.openstack.org/floating-subnet-tags . If multiple subnets match these tags, the first one with available IP addresses is used. If the RHOSP network is configured with sharing disabled, for example, with the --no-share flag used during creation, this option is unsupported. Set the network to share to use this option.

Option	Description
lb-method	<p>The load balancing algorithm used to create the load balancer pool. For the Amphora provider the value can be ROUND_ROBIN, LEAST_CONNECTIONS, or SOURCE_IP. The default value is ROUND_ROBIN.</p> <p>For the OVN provider, only the SOURCE_IP_PORT algorithm is supported.</p> <p>For the Amphora provider, if using the LEAST_CONNECTIONS or SOURCE_IP methods, configure the create-monitor option as true in the cloud-provider-config config map on the openshift-config namespace and ETP:Local on the load-balancer type service to allow balancing algorithm enforcement in the client to service endpoint connections.</p>
lb-provider	Optional. Used to specify the provider of the load balancer, for example, amphora or octavia . Only the Amphora and Octavia providers are supported.
lb-version	Optional. The load balancer API version. Only "v2" is supported.
subnet-id	The ID of the Networking service subnet on which load balancer VIPs are created.
network-id	The ID of the Networking service network on which load balancer VIPs are created. Unnecessary if subnet-id is set.
create-monitor	<p>Whether or not to create a health monitor for the service load balancer. A health monitor is required for services that declare externalTrafficPolicy: Local. The default value is false.</p> <p>This option is unsupported if you use RHOSP earlier than version 17 with the ovn provider.</p>
monitor-delay	The interval in seconds by which probes are sent to members of the load balancer. The default value is 5 .
monitor-max-retries	The number of successful checks that are required to change the operating status of a load balancer member to ONLINE . The valid range is 1 to 10 , and the default value is 1 .

Option	Description
monitor-timeout	The time in seconds that a monitor waits to connect to the back end before it times out. The default value is 3 .
internal-lb	Whether or not to create an internal load balancer without floating IP addresses. The default value is false .
LoadBalancerClass "ClassName"	<p>This is a config section that comprises a set of options:</p> <ul style="list-style-type: none"> ● floating-network-id ● floating-subnet-id ● floating-subnet ● floating-subnet-tags ● network-id ● subnet-id <p>The behavior of these options is the same as that of the identically named options in the load balancer section of the CCM config file.</p> <p>You can set the ClassName value by specifying the service annotation loadbalancer.openstack.org/class.</p>
max-shared-lb	The maximum number of services that can share a load balancer. The default value is 2 .

21.8.2.2. Options that the Operator overrides

The CCM Operator overrides the following options, which you might recognize from configuring RHOSP. Do not configure them yourself. They are included in this document for informational purposes only.

Option	Description
auth-url	The RHOSP Identity service URL. For example, http://128.110.154.166/identity .
os-endpoint-type	The type of endpoint to use from the service catalog.
username	The Identity service user name.
password	The Identity service user password.

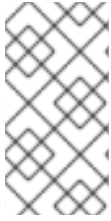
Option	Description
domain-id	The Identity service user domain ID.
domain-name	The Identity service user domain name.
tenant-id	<p>The Identity service project ID. Leave this option unset if you are using Identity service application credentials.</p> <p>In version 3 of the Identity API, which changed the identifier tenant to project, the value of tenant-id is automatically mapped to the project construct in the API.</p>
tenant-name	The Identity service project name.
tenant-domain-id	The Identity service project domain ID.
tenant-domain-name	The Identity service project domain name.
user-domain-id	The Identity service user domain ID.
user-domain-name	The Identity service user domain name.
use-clouds	<p>Whether or not to fetch authorization credentials from a clouds.yaml file. Options set in this section are prioritized over values read from the clouds.yaml file.</p> <p>CCM searches for the file in the following places:</p> <ol style="list-style-type: none"> 1. The value of the clouds-file option. 2. A file path stored in the environment variable OS_CLIENT_CONFIG_FILE. 3. The directory pkg/openstack. 4. The directory ~/.config/openstack. 5. The directory /etc/openstack.
clouds-file	The file path of a clouds.yaml file. It is used if the use-clouds option is set to true .
cloud	The named cloud in the clouds.yaml file that you want to use. It is used if the use-clouds option is set to true .

21.9. UNINSTALLING A CLUSTER ON OPENSTACK

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP).

21.9.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```

$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2

```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

21.10. UNINSTALLING A CLUSTER ON RHOSP FROM YOUR OWN INFRASTRUCTURE

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP) on user-provisioned infrastructure.

21.10.1. Downloading playbook dependencies

The Ansible playbooks that simplify the removal process on user-provisioned infrastructure require several Python modules. On the machine where you will run the process, add the modules' repositories and then download them.

**NOTE**

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:

- a. Register with Red Hat Subscription Manager:

```
$ sudo subscription-manager register # If not done already
```

- b. Pull the latest subscription data:

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

- c. Disable the current repositories:

```
$ sudo subscription-manager repos --disable=* # If not done already
```

- d. Add the required repositories:

```
$ sudo subscription-manager repos \  
--enable=rhel-8-for-x86_64-baseos-rpms \  
--enable=openstack-16-tools-for-rhel-8-x86_64-rpms \  
--enable=ansible-2.9-for-rhel-8-x86_64-rpms \  
--enable=rhel-8-for-x86_64-appstream-rpms
```

2. Install the modules:

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk
```

3. Ensure that the **python** command points to **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

21.10.2. Removing a cluster from RHOSP that uses your own infrastructure

You can remove an OpenShift Container Platform cluster on Red Hat OpenStack Platform (RHOSP) that uses your own infrastructure. To complete the removal process quickly, run several Ansible playbooks.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies."
- You have the playbooks that you used to install the cluster.

- You modified the playbooks that are prefixed with **down-** to reflect any changes that you made to their corresponding installation playbooks. For example, changes to the **bootstrap.yaml** file are reflected in the **down-bootstrap.yaml** file.
- All of the playbooks are in a common directory.

Procedure

1. On a command line, run the playbooks that you downloaded:

```
$ ansible-playbook -i inventory.yaml \  
down-bootstrap.yaml \  
down-control-plane.yaml \  
down-compute-nodes.yaml \  
down-load-balancers.yaml \  
down-network.yaml \  
down-security-groups.yaml
```

2. Remove any DNS record changes you made for the OpenShift Container Platform installation.

OpenShift Container Platform is removed from your infrastructure.

CHAPTER 22. INSTALLING ON RHV

22.1. PREPARING TO INSTALL ON RED HAT VIRTUALIZATION (RHV)

22.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You have a supported combination of versions in the [Support Matrix for OpenShift Container Platform on Red Hat Virtualization \(RHV\)](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

22.1.2. Choosing a method to install OpenShift Container Platform on RHV

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.

22.1.2.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on Red Hat Virtualization (RHV) virtual machines that are provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- **Installing a cluster quickly on RHV** You can quickly install OpenShift Container Platform on RHV virtual machines that the OpenShift Container Platform installation program provisions.
- **Installing a cluster on RHV with customizations** You can install a customized OpenShift Container Platform cluster on installer-provisioned guests on RHV. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available [post-installation](#).

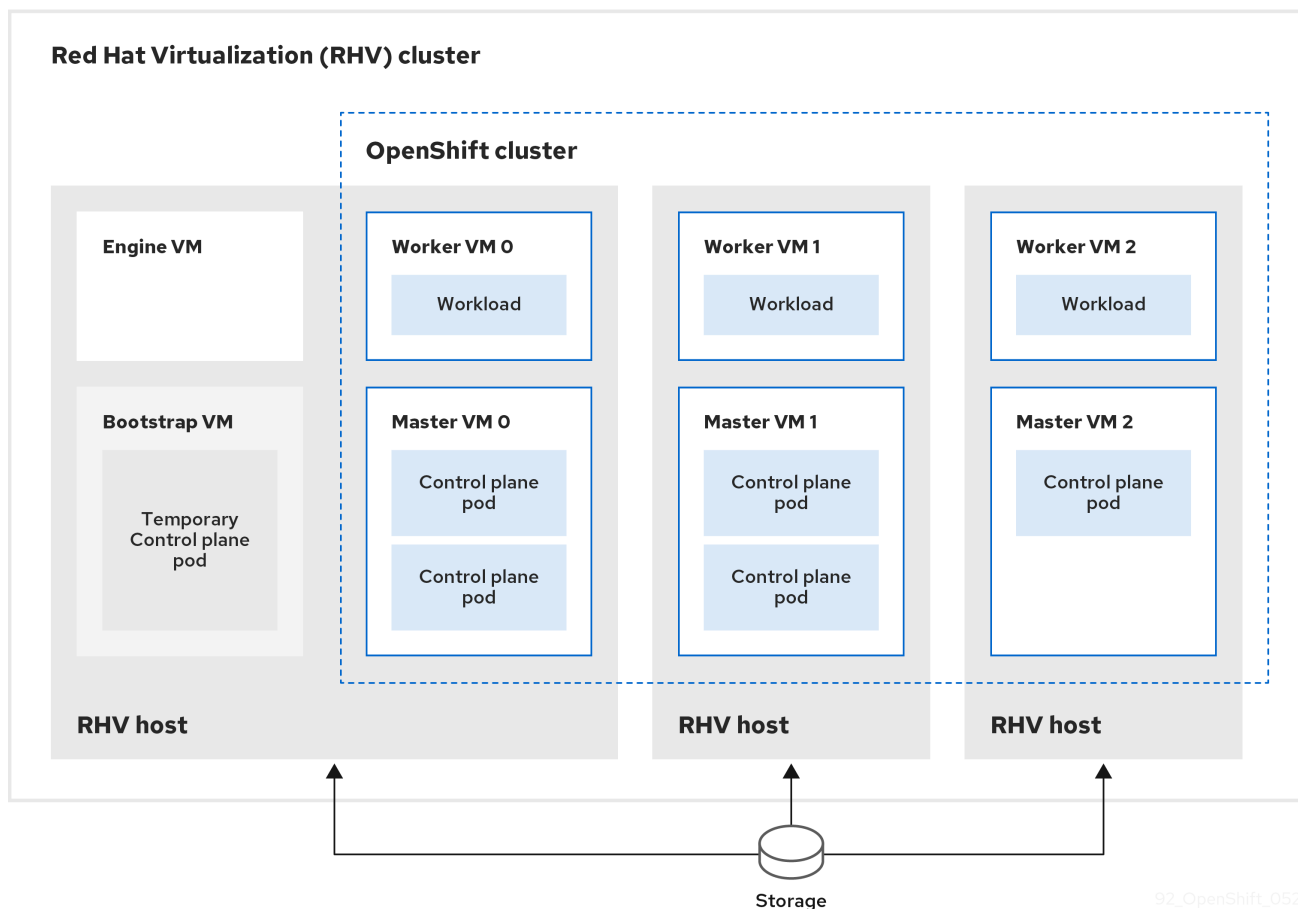
22.1.2.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on RHV virtual machines that you provision, by using one of the following methods:

- **Installing a cluster on RHV with user-provisioned infrastructure** You can install OpenShift Container Platform on RHV virtual machines that you provision. You can use the provided Ansible playbooks to assist with the installation.
- **Installing a cluster on RHV in a restricted network** You can install OpenShift Container Platform on RHV in a restricted or disconnected network by creating an internal mirror of the installation release content. You can use this method to install a user-provisioned cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

22.2. INSTALLING A CLUSTER QUICKLY ON RHV

You can quickly install a default, non-customized, OpenShift Container Platform cluster on a Red Hat Virtualization (RHV) cluster, similar to the one shown in the following diagram.



The installation program uses installer-provisioned infrastructure to automate creating and deploying the cluster.

To install a default cluster, you prepare the environment, run the installation program and answer its prompts. Then, the installation program creates the OpenShift Container Platform cluster.

For an alternative to installing a default cluster, see [Installing a cluster with customizations](#).



NOTE

This installation program is available for Linux and macOS only.

22.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You have a supported combination of versions in the [Support Matrix for OpenShift Container Platform on Red Hat Virtualization \(RHV\)](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

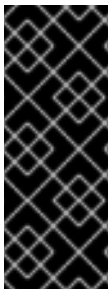
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

22.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

22.2.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform version 4.12 cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on **default** values multiplied by the default number of virtual machines the installation program creates. These resources must be available **in addition to** what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

Requirements

- The RHV version is 4.4.
- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.

- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
 - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
 - 112 GiB RAM or more, including:
 - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
 - 16 GiB or more for each of the three control plane machines which provide the control plane.
 - 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet [these etcd backend performance requirements](#).
- For affinity group support: Three or more hosts in the RHV cluster. If necessary, you can disable affinity groups. For details, see *Example: Removing all affinity groups for a non-production lab setup* in *Installing a cluster on RHV with customizations*
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
 - **DiskOperator**
 - **DiskCreator**
 - **UserTemplateBasedVm**
 - **TemplateOwner**
 - **TemplateCreator**
 - **ClusterAdmin** on the target cluster

**WARNING**

Apply the principle of least privilege: Avoid using an administrator account with **SuperUser** privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary **ovirt-config.yaml** file that might be compromised.

Additional resources

- [Example: Removing all affinity groups for a non-production lab setup](#) .

22.2.4. Verifying the requirements for the RHV environment

Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.

**IMPORTANT**

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check that the RHV version supports installation of OpenShift Container Platform version 4.12.
 - a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select **About**.
 - b. In the window that opens, make a note of the **RHV Software Version**
 - c. Confirm that the RHV version is 4.4. For more information about supported version combinations, see [Support Matrix for OpenShift Container Platform on RHV](#) .
2. Inspect the data center, cluster, and storage.
 - a. In the RHV Administration Portal, click **Compute → Data Centers**.
 - b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
 - c. Click the name of that data center.
 - d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.
 - e. Record the **Domain Name** for use later on.
 - f. Confirm **Free Space** has at least 230 GiB.

- g. Confirm that the storage domain meets [these etcd backend performance requirements](#), which you [can measure by using the fio performance benchmarking tool](#).
 - h. In the data center details, click the **Clusters** tab.
 - i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.
3. Inspect the RHV host resources.
 - a. In the RHV Administration Portal, click **Compute > Clusters**.
 - b. Click the cluster where you plan to install OpenShift Container Platform.
 - c. In the cluster details, click the **Hosts** tab.
 - d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available *exclusively* for the OpenShift Container Platform cluster.
 - e. Record the number of available **Logical CPU Cores** for use later on.
 - f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.
 - g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:
 - 16 GiB required for the bootstrap machine
 - 16 GiB required for each of the three control plane machines
 - 16 GiB for each of the three compute machines
 - h. Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.
 4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager's REST API. From a virtual machine on this network, use curl to reach the RHV Manager's REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1
https://<engine-fqdn>/ovirt-engine/api 2
```

- 1 For **<username>**, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For **<password>**, specify the password for that user name.

- 2 For **<engine-fqdn>**, specify the fully qualified domain name of the RHV environment.

For example:

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

22.2.5. Preparing the network environment on RHV

Configure two static IP addresses for the OpenShift Container Platform cluster and create DNS entries using these addresses.

Procedure

1. Reserve two static IP addresses
 - a. On the network where you plan to install OpenShift Container Platform, identify two static IP addresses that are outside the DHCP lease pool.
 - b. Connect to a host on this network and verify that each of the IP addresses is not in use. For example, use Address Resolution Protocol (ARP) to check that none of the IP addresses have entries:

```
$ arp 10.35.1.19
```

Example output

```
10.35.1.19 (10.35.1.19) -- no entry
```

- c. Reserve two static IP addresses following the standard practices for your network environment.
 - d. Record these IP addresses for future reference.
2. Create DNS entries for the OpenShift Container Platform REST API and apps domain names using this format:

```
api.<cluster-name>.<base-domain> <ip-address> 1  
*.apps.<cluster-name>.<base-domain> <ip-address> 2
```

- 1 For **<cluster-name>**, **<base-domain>**, and **<ip-address>**, specify the cluster name, base domain, and static IP address of your OpenShift Container Platform API.
- 2 Specify the cluster name, base domain, and static IP address of your OpenShift Container Platform apps for Ingress and the load balancer.

For example:

```
api.my-cluster.virtlab.example.com 10.35.1.19  
*.apps.my-cluster.virtlab.example.com 10.35.1.20
```

22.2.6. Installing OpenShift Container Platform on RHV in insecure mode

By default, the installer creates a CA certificate, prompts you for confirmation, and stores the certificate to use during installation. You do not need to create or install one manually.

Although it is not recommended, you can override this functionality and install OpenShift Container Platform without verifying a certificate by installing OpenShift Container Platform on RHV in **insecure** mode.



WARNING

Installing in **insecure** mode is not recommended, because it enables a potential attacker to perform a Man-in-the-Middle attack and capture sensitive credentials on the network.

Procedure

1. Create a file named `~/ovirt/ovirt-config.yaml`.
2. Add the following content to **ovirt-config.yaml**:

```
ovirt_url: https://ovirt.example.com/ovirt-engine/api 1
ovirt_fqdn: ovirt.example.com 2
ovirt_pem_url: ""
ovirt_username: ocpadmin@internal
ovirt_password: super-secret-password 3
ovirt_insecure: true
```

- 1 Specify the hostname or address of your oVirt engine.
- 2 Specify the fully qualified domain name of your oVirt engine.
- 3 Specify the admin password for your oVirt engine.

3. Run the installer.

22.2.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

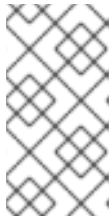
Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

22.2.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

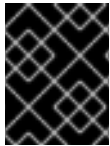
```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities,

including Quay.io, which serves the container images for OpenShift Container Platform components.

22.2.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Open the **ovirt-imageio** port to the Manager from the machine running the installer. By default, the port is **54322**.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Respond to the installation program prompts.
 - a. Optional: For **SSH Public Key**, select a password-less public key, such as `~/.ssh/id_rsa.pub`. This key authenticates connections with the new OpenShift Container Platform cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, select an SSH key that your **ssh-agent** process uses.

- b. For **Platform**, select **ovirt**.
- c. For **Engine FQDN[:PORT]**, enter the fully qualified domain name (FQDN) of the RHV environment.
For example:


```
rhv-env.virtlab.example.com:443
```
- d. The installation program automatically generates a CA certificate. For **Would you like to use the above certificate to connect to the Manager?**, answer **y** or **N**. If you answer **N**, you must install OpenShift Container Platform in insecure mode.
- e. For **Engine username**, enter the user name and profile of the RHV administrator using this format:


```
<username>@<profile> 1
```

1 For **<username>**, specify the user name of an RHV administrator. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For example: **admin@internal**.
- f. For **Engine password**, enter the RHV admin password.
- g. For **Cluster**, select the RHV cluster for installing OpenShift Container Platform.
- h. For **Storage domain**, select the storage domain for installing OpenShift Container Platform.
- i. For **Network**, select a virtual network that has access to the RHV Manager REST API.
- j. For **Internal API Virtual IP**, enter the static IP address you set aside for the cluster's REST API.
- k. For **Ingress virtual IP**, enter the static IP address you reserved for the wildcard apps domain.
- l. For **Base Domain**, enter the base domain of the OpenShift Container Platform cluster. If this cluster is exposed to the outside world, this must be a valid domain recognized by DNS infrastructure. For example, enter: **virtlab.example.com**
- m. For **Cluster Name**, enter the name of the cluster. For example, **my-cluster**. Use cluster name from the externally registered/resolvable DNS entries you created for the OpenShift Container Platform REST API and apps domain names. The installation program also gives this name to the cluster in the RHV environment.
- n. For **Pull Secret**, copy the pull secret from the **pull-secret.txt** file you downloaded earlier and paste it here. You can also get a copy of the same [pull secret from the Red Hat OpenShift Cluster Manager](#).

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

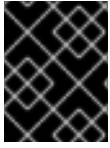
- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You have completed the steps required to install the cluster. The remaining steps show you how to verify the cluster and troubleshoot the installation.

22.2.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

■

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

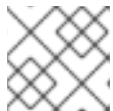
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

To learn more, see [Getting started with the OpenShift CLI](#).

22.2.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

22.2.12. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

3. View your cluster's version:

```
$ oc get clusterversion
```

4. View your Operators' status:

```
$ oc get clusteroperator
```

5. View all running pods in the cluster:

```
-
```

```
$ oc get pods -A
```

Troubleshooting

If the installation fails, the installation program times out and displays an error message. To learn more, see [Troubleshooting installation issues](#).

22.2.13. Accessing the OpenShift Container Platform web console on RHV

After the OpenShift Container Platform cluster initializes, you can log in to the OpenShift Container Platform web console.

Procedure

1. Optional: In the Red Hat Virtualization (RHV) Administration Portal, open **Compute** → **Cluster**.
2. Verify that the installation program creates the virtual machines.
3. Return to the command line where the installation program is running. When the installation program finishes, it displays the user name and temporary password for logging into the OpenShift Container Platform web console.
4. In a browser, open the URL of the OpenShift Container Platform web console. The URL uses this format:

```
console-openshift-console.apps.<clustername>.<basedomain> 1
```

- 1** For **<clustername>.<basedomain>**, specify the cluster name and base domain.

For example:

```
console-openshift-console.apps.my-cluster.virtlab.example.com
```

22.2.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

22.2.15. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)

Here are some common issues you might encounter, along with proposed causes and solutions.

22.2.15.1. CPU load increases and nodes go into a **Not Ready** state

- **Symptom:** CPU load increases significantly and nodes start going into a **Not Ready** state.
- **Cause:** The storage domain latency might be too high, especially for control plane nodes.
- **Solution:**
Make the nodes ready again by restarting the kubelet service:

```
$ systemctl restart kubelet
```

Inspect the OpenShift Container Platform metrics service, which automatically gathers and reports on some valuable data such as the etcd disk sync duration. If the cluster is operational, use this data to help determine whether storage latency or throughput is the root issue. If so, consider using a storage resource that has lower latency and higher throughput.

To get raw metrics, enter the following command as kubeadmin or user with cluster-admin privileges:

```
$ oc get --insecure-skip-tls-verify --server=https://localhost:<port> --raw=/metrics
```

To learn more, see [Exploring Application Endpoints for the purposes of Debugging with OpenShift 4.x](#)

22.2.15.2. Trouble connecting the OpenShift Container Platform cluster API

- **Symptom:** The installation program completes but the OpenShift Container Platform cluster API is not available. The bootstrap virtual machine remains up after the bootstrap process is complete. When you enter the following command, the response will time out.

```
$ oc login -u kubeadmin -p *** <apiurl>
```

- **Cause:** The bootstrap VM was not deleted by the installation program and has not released the cluster's API IP address.
- **Solution:** Use the **wait-for** subcommand to be notified when the bootstrap process is complete:

```
$ ./openshift-install wait-for bootstrap-complete
```

When the bootstrap process is complete, delete the bootstrap virtual machine:

```
$ ./openshift-install destroy bootstrap
```

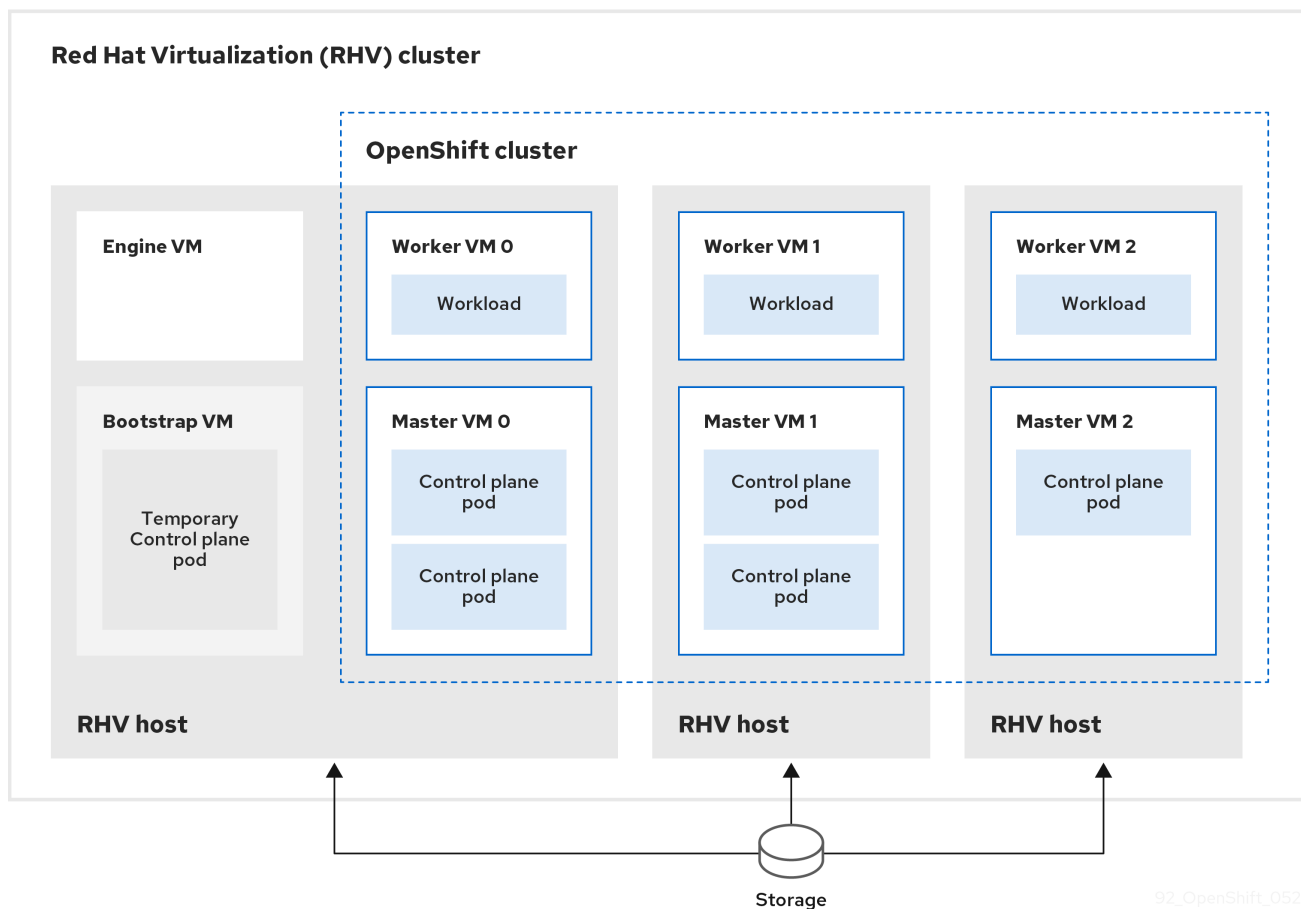
22.2.16. Post-installation tasks

After the OpenShift Container Platform cluster initializes, you can perform the following tasks.

- Optional: After deployment, add or replace SSH keys using the Machine Config Operator (MCO) in OpenShift Container Platform.
- Optional: Remove the **kubeadmin** user. Instead, use the authentication provider to create a user with cluster-admin privileges.

22.3. INSTALLING A CLUSTER ON RHV WITH CUSTOMIZATIONS

You can customize and install an OpenShift Container Platform cluster on Red Hat Virtualization (RHV), similar to the one shown in the following diagram.



92_OpenShift_0520

The installation program uses installer-provisioned infrastructure to automate creating and deploying the cluster.

To install a customized cluster, you prepare the environment and perform the following steps:

1. Create an installation configuration file, the **install-config.yaml** file, by running the installation program and answering its prompts.
2. Inspect and modify parameters in the **install-config.yaml** file.
3. Make a working copy of the **install-config.yaml** file.
4. Run the installation program with a copy of the **install-config.yaml** file.

Then, the installation program creates the OpenShift Container Platform cluster.

For an alternative to installing a customized cluster, see [Installing a default cluster](#).



NOTE

This installation program is available for Linux and macOS only.

22.3.1. Prerequisites

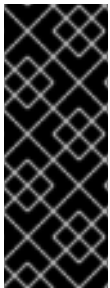
- You reviewed details about the [OpenShift Container Platform installation and update processes](#).
- You have a supported combination of versions in the [Support Matrix for OpenShift Container Platform on Red Hat Virtualization \(RHV\)](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

22.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

22.3.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform version 4.12 cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on **default** values multiplied by the default number of virtual machines the installation program creates. These resources must be available **in addition to** what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

Requirements

- The RHV version is 4.4.
- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
 - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
 - 112 GiB RAM or more, including:
 - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
 - 16 GiB or more for each of the three control plane machines which provide the control plane.
 - 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet [these etcd backend performance requirements](#).
- For affinity group support:

One physical machine per worker or control plane. Workers and control planes can be on the same physical machine. For example, if you have three workers and three control planes, you need three physical machines. If you have four workers and three control planes, you need four physical machines.

 - For hard anti-affinity (default): A minimum of three physical machines. For more than three worker nodes, one physical machine per worker or control plane. Workers and control planes can be on the same physical machine.
 - For custom affinity groups: Ensure that the resources are appropriate for the affinity group rules that you define.
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
 - **DiskOperator**
 - **DiskCreator**

- **UserTemplateBasedVm**
- **TemplateOwner**
- **TemplateCreator**
- **ClusterAdmin** on the target cluster

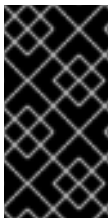


WARNING

Apply the principle of least privilege: Avoid using an administrator account with **SuperUser** privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary **ovirt-config.yaml** file that might be compromised.

22.3.4. Verifying the requirements for the RHV environment

Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.



IMPORTANT

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check that the RHV version supports installation of OpenShift Container Platform version 4.12.
 - a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select **About**.
 - b. In the window that opens, make a note of the **RHV Software Version**
 - c. Confirm that the RHV version is 4.4. For more information about supported version combinations, see [Support Matrix for OpenShift Container Platform on RHV](#).
2. Inspect the data center, cluster, and storage.
 - a. In the RHV Administration Portal, click **Compute → Data Centers**.
 - b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
 - c. Click the name of that data center.
 - d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.

- e. Record the **Domain Name** for use later on.
 - f. Confirm **Free Space** has at least 230 GiB.
 - g. Confirm that the storage domain meets [these etcd backend performance requirements](#), which you [can measure by using the fio performance benchmarking tool](#).
 - h. In the data center details, click the **Clusters** tab.
 - i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.
3. Inspect the RHV host resources.
- a. In the RHV Administration Portal, click **Compute > Clusters**.
 - b. Click the cluster where you plan to install OpenShift Container Platform.
 - c. In the cluster details, click the **Hosts** tab.
 - d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available *exclusively* for the OpenShift Container Platform cluster.
 - e. Record the number of available **Logical CPU Cores** for use later on.
 - f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.
 - g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:
 - 16 GiB required for the bootstrap machine
 - 16 GiB required for each of the three control plane machines
 - 16 GiB for each of the three compute machines
 - h. Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.
4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager's REST API. From a virtual machine on this network, use curl to reach the RHV Manager's REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1  
https://<engine-fqdn>/ovirt-engine/api 2
```

1 For **<username>**, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For **<password>**, specify the password for that user name.

2 For **<engine-fqdn>**, specify the fully qualified domain name of the RHV environment.

For example:

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

22.3.5. Preparing the network environment on RHV

Configure two static IP addresses for the OpenShift Container Platform cluster and create DNS entries using these addresses.

Procedure

1. Reserve two static IP addresses
 - a. On the network where you plan to install OpenShift Container Platform, identify two static IP addresses that are outside the DHCP lease pool.
 - b. Connect to a host on this network and verify that each of the IP addresses is not in use. For example, use Address Resolution Protocol (ARP) to check that none of the IP addresses have entries:

```
$ arp 10.35.1.19
```

Example output

```
10.35.1.19 (10.35.1.19) -- no entry
```

- c. Reserve two static IP addresses following the standard practices for your network environment.
 - d. Record these IP addresses for future reference.
2. Create DNS entries for the OpenShift Container Platform REST API and apps domain names using this format:

```
api.<cluster-name>.<base-domain> <ip-address> 1
*.apps.<cluster-name>.<base-domain> <ip-address> 2
```

- 1 For **<cluster-name>**, **<base-domain>**, and **<ip-address>**, specify the cluster name, base domain, and static IP address of your OpenShift Container Platform API.
- 2 Specify the cluster name, base domain, and static IP address of your OpenShift Container Platform apps for Ingress and the load balancer.

For example:

```
api.my-cluster.virtlab.example.com 10.35.1.19
*.apps.my-cluster.virtlab.example.com 10.35.1.20
```

22.3.6. Installing OpenShift Container Platform on RHV in insecure mode

By default, the installer creates a CA certificate, prompts you for confirmation, and stores the certificate to use during installation. You do not need to create or install one manually.

Although it is not recommended, you can override this functionality and install OpenShift Container Platform without verifying a certificate by installing OpenShift Container Platform on RHV in **insecure** mode.



WARNING

Installing in **insecure** mode is not recommended, because it enables a potential attacker to perform a Man-in-the-Middle attack and capture sensitive credentials on the network.

Procedure

1. Create a file named `~/ovirt/ovirt-config.yaml`.
2. Add the following content to **ovirt-config.yaml**:

```
ovirt_url: https://ovirt.example.com/ovirt-engine/api 1
ovirt_fqdn: ovirt.example.com 2
ovirt_pem_url: ""
ovirt_username: ocpadmin@internal
ovirt_password: super-secret-password 3
ovirt_insecure: true
```

- 1 Specify the hostname or address of your oVirt engine.
- 2 Specify the fully qualified domain name of your oVirt engine.
- 3 Specify the admin password for your oVirt engine.

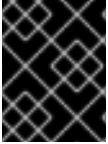
3. Run the installer.

22.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

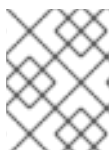
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

22.3.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

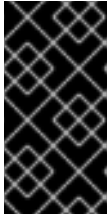
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

22.3.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat Virtualization (RHV).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. Respond to the installation program prompts.

- i. For **SSH Public Key**, select a password-less public key, such as `~/.ssh/id_rsa.pub`. This key authenticates connections with the new OpenShift Container Platform cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, select an SSH key that your **ssh-agent** process uses.

- ii. For **Platform**, select **ovirt**.
- iii. For **Enter oVirt's API endpoint URL**, enter the URL of the RHV API using this format:

```
https://<engine-fqdn>/ovirt-engine/api 1
```

- 1** For **<engine-fqdn>**, specify the fully qualified domain name of the RHV environment.

For example:

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

- iv. For **Is the oVirt CA trusted locally?**, enter **Yes**, because you have already set up a CA certificate. Otherwise, enter **No**.
- v. For **oVirt's CA bundle**, if you entered **Yes** for the preceding question, copy the certificate content from `/etc/pki/ca-trust/source/anchors/ca.pem` and paste it here. Then, press **Enter** twice. Otherwise, if you entered **No** for the preceding question, this question does not appear.
- vi. For **oVirt engine username**, enter the user name and profile of the RHV administrator using this format:

```
<username>@<profile> 1
```

- 1** For **<username>**, specify the user name of an RHV administrator. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. Together, the user name and profile should look similar to this example:

```
ocpadmin@internal
```

- vii. For **oVirt engine password**, enter the RHV admin password.
- viii. For **oVirt cluster**, select the cluster for installing OpenShift Container Platform.
- ix. For **oVirt storage domain**, select the storage domain for installing OpenShift Container Platform.
- x. For **oVirt network**, select a virtual network that has access to the RHV Manager REST API.

- xi. For **Internal API Virtual IP**, enter the static IP address you set aside for the cluster's REST API.
 - xii. For **Ingress virtual IP**, enter the static IP address you reserved for the wildcard apps domain.
 - xiii. For **Base Domain**, enter the base domain of the OpenShift Container Platform cluster. If this cluster is exposed to the outside world, this must be a valid domain recognized by DNS infrastructure. For example, enter: **virtlab.example.com**
 - xiv. For **Cluster Name**, enter the name of the cluster. For example, **my-cluster**. Use cluster name from the externally registered/resolvable DNS entries you created for the OpenShift Container Platform REST API and apps domain names. The installation program also gives this name to the cluster in the RHV environment.
 - xv. For **Pull Secret**, copy the pull secret from the **pull-secret.txt** file you downloaded earlier and paste it here. You can also get a copy of the same [pull secret from the Red Hat OpenShift Cluster Manager](#).
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.



NOTE

If you have any intermediate CA certificates on the Manager, verify that the certificates appear in the **ovirt-config.yaml** file and the **install-config.yaml** file. If they do not appear, add them as follows:

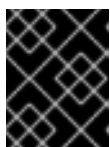
1. In the `~/ovirt/ovirt-config.yaml` file:

```
[ovirt_ca_bundle]: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA>
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  <INTERMEDIATE_CA>
  -----END CERTIFICATE-----
```

2. In the **install-config.yaml** file:

```
[additionalTrustBundle]: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA>
  -----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----
  <INTERMEDIATE_CA>
  -----END CERTIFICATE-----
```

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

22.3.9.1. Example `install-config.yaml` files for Red Hat Virtualization (RHV)

You can customize the OpenShift Container Platform cluster the installation program creates by changing the parameters and parameter values in the `install-config.yaml` file.

The following examples are specific to installing OpenShift Container Platform on RHV.

`install-config.yaml` is located in `<installation_directory>`, which you specified when you ran the following command.

```
$ ./openshift-install create install-config --dir <installation_directory>
```



NOTE

- These example files are provided for reference only. You must obtain your `install-config.yaml` file by using the installation program.
- Changing the `install-config.yaml` file can increase the resources your cluster requires. Verify that your RHV environment has those additional resources. Otherwise, the installation or cluster will fail.

Example default `install-config.yaml` file

```
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    ovirt:
      sparse: false 1
      format: raw 2
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    ovirt:
      sparse: false 3
      format: raw 4
  replicas: 3
metadata:
  creationTimestamp: null
  name: my-cluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 5
serviceNetwork:
```

```

- 172.30.0.0/16
platform:
  ovirt:
    api_vips:
      - 10.0.0.10
    ingress_vips:
      - 10.0.0.11
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
publish: External
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...

```

- 1 3** Setting this option to **false** enables preallocation of disks. The default is **true**. Setting **sparse** to **true** with **format** set to **raw** is not available for block storage domains. The **raw** format writes the entire virtual disk to the underlying physical disk.



NOTE

Preallocating disks on file storage domains writes zeroes to the file. This might not actually preallocate disks depending on the underlying storage.

- 2 4** Can be set to **cow** or **raw**. The default is **cow**. The **cow** format is optimized for virtual machines.
- 5** The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.



NOTE

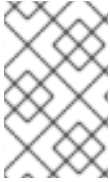
In OpenShift Container Platform 4.12 and later, the **api_vip** and **ingress_vip** configuration settings are deprecated. Instead, use a list format to enter values in the **api_vips** and **ingress_vips** configuration settings.

Example minimal `install-config.yaml` file

```

apiVersion: v1
baseDomain: example.com
metadata:
  name: test-cluster
platform:
  ovirt:
    api_vips:
      - 10.46.8.230
    ingress_vips:
      - 10.46.8.232
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...

```

**NOTE**

In OpenShift Container Platform 4.12 and later, the **api_vip** and **ingress_vip** configuration settings are deprecated. Instead, use a list format to enter values in the **api_vips** and **ingress_vips** configuration settings.

Example Custom machine pools in arinstall-config.yaml file

```

apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform:
    ovirt:
      cpu:
        cores: 4
        sockets: 2
      memoryMB: 65536
      osDisk:
        sizeGB: 100
      vmType: server
  replicas: 3
compute:
- name: worker
  platform:
    ovirt:
      cpu:
        cores: 4
        sockets: 4
      memoryMB: 65536
      osDisk:
        sizeGB: 200
      vmType: server
  replicas: 5
metadata:
  name: test-cluster
platform:
  ovirt:
    api_vips:
      - 10.46.8.230
    ingress_vips:
      - 10.46.8.232
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...

```

**NOTE**

In OpenShift Container Platform 4.12 and later, the **api_vip** and **ingress_vip** configuration settings are deprecated. Instead, use a list format to enter values in the **api_vips** and **ingress_vips** configuration settings.

Example non-enforcing affinity group

It is recommended to add a non-enforcing affinity group to distribute the control plane and workers, if possible, to use as much of the cluster as possible.

```
platform:
  ovirt:
    affinityGroups:
      - description: AffinityGroup to place each compute machine on a separate host
        enforcing: true
        name: compute
        priority: 3
      - description: AffinityGroup to place each control plane machine on a separate host
        enforcing: true
        name: controlplane
        priority: 5
      - description: AffinityGroup to place worker nodes and control plane nodes on separate hosts
        enforcing: false
        name: openshift
        priority: 5
  compute:
    - architecture: amd64
      hyperthreading: Enabled
      name: worker
      platform:
        ovirt:
          affinityGroupsNames:
            - compute
            - openshift
      replicas: 3
  controlPlane:
    architecture: amd64
    hyperthreading: Enabled
    name: master
    platform:
      ovirt:
        affinityGroupsNames:
          - controlplane
          - openshift
    replicas: 3
```

Example removing all affinity groups for a non-production lab setup

For non-production lab setups, you must remove all affinity groups to concentrate the OpenShift Container Platform cluster on the few hosts you have.

```
platform:
  ovirt:
    affinityGroups: []
  compute:
    - architecture: amd64
      hyperthreading: Enabled
      name: worker
      platform:
        ovirt:
          affinityGroupsNames: []
    replicas: 3
```

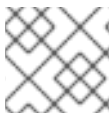
```

controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    ovirt:
      affinityGroupsNames: []
  replicas: 3

```

22.3.9.2. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

22.3.9.2.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 22.1. Required parameters

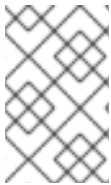
Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .

Parameter	Description	Values
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters, hyphens (-), and periods (.), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

22.3.9.2.2. Network configuration parameters


You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.


Only IPv4 addresses are supported.

**NOTE**

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 22.2. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .


Parameter	Description	Values
networking.serviceNetwork	<p>The IP address block for services. The default value is 172.30.0.0/16.</p> <p>The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.</p>	<p>An array with an IP address block in CIDR format. For example:</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>The IP address blocks for machines.</p> <p>If you specify multiple IP address blocks, the blocks must not overlap.</p>	<p>An array of objects. For example:</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</p>	<p>An IP network block in CIDR notation.</p> <p>For example, 10.0.0.0/16.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p> </div> </div>

22.3.9.2.3. Optional configuration parameters


Optional installation configuration parameters are described in the following table:



Table 22.3. Optional parameters

Parameter	Description	Values
additionalTrustBundle	<p>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</p>	String



Parameter	Description	Values
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects. For details, see the "Additional RHV parameters for machine pools" table.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled

Parameter	Description	Values
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects. For details, see the "Additional RHV parameters for machine pools" table.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String

Parameter	Description	Values
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p>	false or true

Parameter	Description	Values
	<p data-bbox="671 107 863 141">IMPORTANT</p> <p data-bbox="671 181 932 909">To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <p data-bbox="671 965 762 999">NOTE</p> <p data-bbox="671 1032 922 1151">If you are using Azure File storage, you cannot enable FIPS mode.</p>	
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings

Parameter	Description	Values
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div>	For example, sshKey: ssh-ed25519 AAAA...


22.3.9.2.4. Additional Red Hat Virtualization (RHV) configuration parameters

Additional RHV configuration parameters are described in the following table:

Table 22.4. Additional Red Hat Virtualization (RHV) parameters for clusters

Parameter	Description	Values
platform.ovirt.ovirt_cluster_id	Required. The Cluster where the VMs will be created.	String. For example: 68833f9f-e89c-4891-b768-e2ba0815b76b
platform.ovirt.ovirt_storage_domain_id	Required. The Storage Domain ID where the VM disks will be created.	String. For example: ed7b0f4e-0e96-492a-8fff-279213ee1468
platform.ovirt.ovirt_network_name	Required. The network name where the VM nics will be created.	String. For example: ocpcluster

Parameter	Description	Values
platform.ovirt.vnicProfileID	Required. The vNIC profile ID of the VM network interfaces. This can be inferred if the cluster network has a single profile.	String. For example: 3fa86930-0be5-4052-b667-b79f0a729692
platform.ovirt.api_vips	<p>Required. An IP address on the machine network that will be assigned to the API virtual IP (VIP). You can access the OpenShift API at this endpoint. For dual-stack networks, assign up to two IP addresses. The primary IP address must be from the IPv4 network.</p> <div style="display: flex; align-items: flex-start;"> <div style="border: 1px solid black; width: 20px; height: 20px; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the api_vip configuration setting is deprecated. Instead, use a list format to enter a value in the api_vips configuration setting. The order of the list indicates the primary and secondary VIP address for each service.</p> </div> </div>	String. Example: 10.46.8.230

Parameter	Description	Values
platform.ovirt.ingress_vips	<p>Required. An IP address on the machine network that will be assigned to the Ingress virtual IP (VIP). For dual-stack networks, assign up to two IP addresses. The primary IP address must be from the IPv4 network.</p> <div style="display: flex; align-items: center;">  <div> <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingress_vip configuration setting is deprecated. Instead, use a list format to enter a value in the ingress_vips configuration setting. The order of the list indicates the primary and secondary VIP address for each service.</p> </div> </div>	String. Example: 10.46.8.232
platform.ovirt.affinityGroups	Optional. A list of affinity groups to create during the installation process.	List of objects.
platform.ovirt.affinityGroups.description	Required if you include platform.ovirt.affinityGroups . A description of the affinity group.	String. Example: AffinityGroup for spreading each compute machine to a different host
platform.ovirt.affinityGroups.enforcing	Required if you include platform.ovirt.affinityGroups . When set to true , RHV does not provision any machines if not enough hardware nodes are available. When set to false , RHV does provision machines even if not enough hardware nodes are available, resulting in multiple virtual machines being hosted on the same physical machine.	String. Example: true
platform.ovirt.affinityGroups.name	Required if you include platform.ovirt.affinityGroups . The name of the affinity group.	String. Example: compute


Parameter	Description	Values
platform.ovirt.affinityGroups.priority	Required if you include platform.ovirt.affinityGroups . The priority given to an affinity group when platform.ovirt.affinityGroups.enforcing = false . RHV applies affinity groups in the order of priority, where a greater number takes precedence over a lesser one. If multiple affinity groups have the same priority, the order in which they are applied is not guaranteed.	Integer. Example: 3

22.3.9.2.5. Additional RHV parameters for machine pools

Additional RHV configuration parameters for machine pools are described in the following table:

Table 22.5. Additional RHV parameters for machine pools

Parameter	Description	Values
<machine-pool>.platform.ovirt.cpu	Optional. Defines the CPU of the VM.	Object
<machine-pool>.platform.ovirt.cpu.cores	Required if you use <machine-pool>.platform.ovirt.cpu . The number of cores. Total virtual CPUs (vCPUs) is cores * sockets.	Integer
<machine-pool>.platform.ovirt.cpu.sockets	Required if you use <machine-pool>.platform.ovirt.cpu . The number of sockets per core. Total virtual CPUs (vCPUs) is cores * sockets.	Integer
<machine-pool>.platform.ovirt.memoryMB	Optional. Memory of the VM in MiB.	Integer
<machine-pool>.platform.ovirt.osDisk	Optional. Defines the first and bootable disk of the VM.	String
<machine-pool>.platform.ovirt.osDisk.sizeGB	Required if you use <machine-pool>.platform.ovirt.osDisk . Size of the disk in GiB.	Number

Parameter	Description	Values
<p><machine-pool>.platform.ovirt.vmPid</p>	<p>Optional. The VM workload type, such as high-performance, server, or desktop. By default, control plane nodes use high-performance, and worker nodes use server. For details, see Explanation of Settings in the New Virtual Machine and Edit Virtual Machine Windows and Configuring High Performance Virtual Machines, Templates, and Pools in the <i>Virtual Machine Management Guide</i>.</p> <div data-bbox="485 656 595 1160" style="border: 1px solid black; padding: 5px; width: fit-content;">  </div> <p>NOTE</p> <p>high_performance improves performance on the VM, but there are limitations. For example, you cannot access the VM with a graphical console. For more information, see Configuring High Performance Virtual Machines, Templates, and Pools in the <i>Virtual Machine Management Guide</i>.</p>	<p>String</p>

Parameter	Description	Values
<code><machine-pool>.platform.ovirt.affinityGroupNames</code>	<p>Optional. A list of affinity group names that should be applied to the virtual machines. The affinity groups must exist in RHV, or be created during installation as described in <i>Additional RHV parameters for clusters</i> in this topic. This entry can be empty.</p> <p>Example with two affinity groups</p> <p>This example defines two affinity groups, named compute and clusterWideNonEnforcing:</p> <pre><machine-pool>: platform: ovirt: affinityGroupNames: - compute - clusterWideNonEnforcing</pre> <p>This example defines no affinity groups:</p> <pre><machine-pool>: platform: ovirt: affinityGroupNames: []</pre>	String
<code><machine-pool>.platform.ovirt.AutoPinningPolicy</code>	<p>Optional. <code>AutoPinningPolicy</code> defines the policy to automatically set the CPU and NUMA settings, including pinning to the host for the instance. When the field is omitted, the default is none. Supported values: none, resize_and_pin. For more information, see Setting NUMA Nodes in the <i>Virtual Machine Management Guide</i>.</p>	String
<code><machine-pool>.platform.ovirt.hugepages</code>	<p>Optional. <code>Hugepages</code> is the size in KiB for defining hugepages in a VM. Supported values: 2048 or 1048576. For more information, see Configuring Huge Pages in the <i>Virtual Machine Management Guide</i>.</p>	Integer

**NOTE**

You can replace `<machine-pool>` with **controlPlane** or **compute**.

22.3.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Open the **ovirt-imageio** port to the Manager from the machine running the installer. By default, the port is **54322**.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

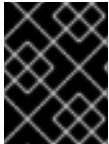
**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

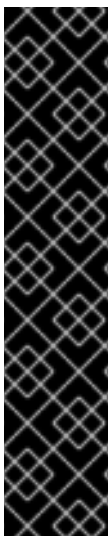
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

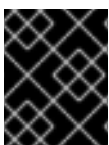
- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You have completed the steps required to install the cluster. The remaining steps show you how to verify the cluster and troubleshoot the installation.

22.3.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

22.3.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

■


```
system:admin
```

To learn more, see [Getting started with the OpenShift CLI](#).

22.3.13. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

3. View your cluster's version:

```
$ oc get clusterversion
```

4. View your Operators' status:

```
$ oc get clusteroperator
```

5. View all running pods in the cluster:

```
$ oc get pods -A
```

Troubleshooting

If the installation fails, the installation program times out and displays an error message. To learn more, see [Troubleshooting installation issues](#).

22.3.14. Accessing the OpenShift Container Platform web console on RHV

After the OpenShift Container Platform cluster initializes, you can log in to the OpenShift Container Platform web console.

Procedure

1. Optional: In the Red Hat Virtualization (RHV) Administration Portal, open **Compute → Cluster**.
2. Verify that the installation program creates the virtual machines.

- Return to the command line where the installation program is running. When the installation program finishes, it displays the user name and temporary password for logging into the OpenShift Container Platform web console.
- In a browser, open the URL of the OpenShift Container Platform web console. The URL uses this format:

```
console-openshift-console.apps.<clustername>.<basedomain> 1
```

- For **<clustername>.<basedomain>**, specify the cluster name and base domain.

For example:

```
console-openshift-console.apps.my-cluster.virtlab.example.com
```

22.3.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

22.3.16. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)

Here are some common issues you might encounter, along with proposed causes and solutions.

22.3.16.1. CPU load increases and nodes go into a **Not Ready** state

- Symptom:** CPU load increases significantly and nodes start going into a **Not Ready** state.
- Cause:** The storage domain latency might be too high, especially for control plane nodes.
- Solution:**
Make the nodes ready again by restarting the kubelet service:

```
$ systemctl restart kubelet
```

Inspect the OpenShift Container Platform metrics service, which automatically gathers and reports on some valuable data such as the etcd disk sync duration. If the cluster is operational, use this data to help determine whether storage latency or throughput is the root issue. If so, consider using a storage resource that has lower latency and higher throughput.

To get raw metrics, enter the following command as kubeadmin or user with cluster-admin privileges:

```
$ oc get --insecure-skip-tls-verify --server=https://localhost:<port> --raw=/metrics
```

To learn more, see [Exploring Application Endpoints for the purposes of Debugging with OpenShift 4.x](#)

22.3.16.2. Trouble connecting the OpenShift Container Platform cluster API

- **Symptom:** The installation program completes but the OpenShift Container Platform cluster API is not available. The bootstrap virtual machine remains up after the bootstrap process is complete. When you enter the following command, the response will time out.

```
$ oc login -u kubeadmin -p *** <apiurl>
```

- **Cause:** The bootstrap VM was not deleted by the installation program and has not released the cluster's API IP address.
- **Solution:** Use the **wait-for** subcommand to be notified when the bootstrap process is complete:

```
$ ./openshift-install wait-for bootstrap-complete
```

When the bootstrap process is complete, delete the bootstrap virtual machine:

```
$ ./openshift-install destroy bootstrap
```

22.3.17. Post-installation tasks

After the OpenShift Container Platform cluster initializes, you can perform the following tasks.

- Optional: After deployment, add or replace SSH keys using the Machine Config Operator (MCO) in OpenShift Container Platform.
- Optional: Remove the **kubeadmin** user. Instead, use the authentication provider to create a user with cluster-admin privileges.

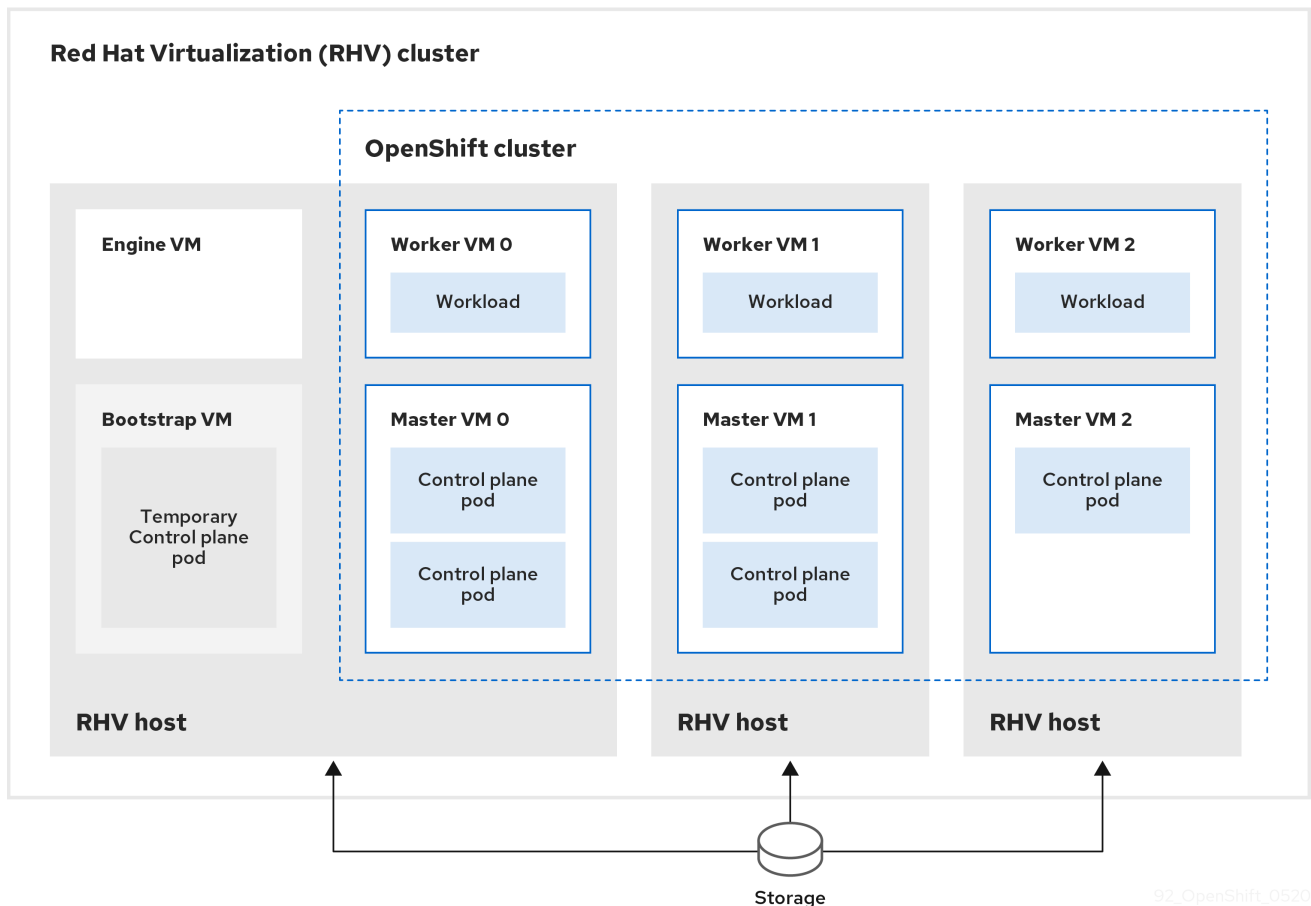
22.3.18. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .

22.4. INSTALLING A CLUSTER ON RHV WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a customized OpenShift Container Platform cluster on Red Hat Virtualization (RHV) and other infrastructure that you provide. The OpenShift Container Platform documentation uses the term *user-provisioned infrastructure* to refer to this infrastructure type.

The following diagram shows an example of a potential OpenShift Container Platform cluster running on a RHV cluster.



92_OpenShift_0520

The RHV hosts run virtual machines that contain both control plane and compute pods. One of the hosts also runs a Manager virtual machine and a bootstrap virtual machine that contains a temporary control plane pod.]

22.4.1. Prerequisites

The following items are required to install an OpenShift Container Platform cluster on a RHV environment.

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You have a supported combination of versions in the [Support Matrix for OpenShift Container Platform on Red Hat Virtualization \(RHV\)](#).

22.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

22.4.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform version 4.12 cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on **default** values multiplied by the default number of virtual machines the installation program creates. These resources must be available **in addition to** what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

Requirements

- The RHV version is 4.4.
- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
 - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
 - 112 GiB RAM or more, including:
 - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
 - 16 GiB or more for each of the three control plane machines which provide the control plane.

- 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet [these etcd backend performance requirements](#).
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
 - **DiskOperator**
 - **DiskCreator**
 - **UserTemplateBasedVm**
 - **TemplateOwner**
 - **TemplateCreator**
 - **ClusterAdmin** on the target cluster

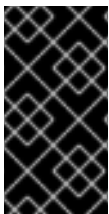


WARNING

Apply the principle of least privilege: Avoid using an administrator account with **SuperUser** privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary **ovirt-config.yaml** file that might be compromised.

22.4.4. Verifying the requirements for the RHV environment

Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.



IMPORTANT

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check that the RHV version supports installation of OpenShift Container Platform version 4.12.
 - a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select **About**.
 - b. In the window that opens, make a note of the **RHV Software Version**
 - c. Confirm that the RHV version is 4.4. For more information about supported version combinations, see [Support Matrix for OpenShift Container Platform on RHV](#).
2. Inspect the data center, cluster, and storage.
 - a. In the RHV Administration Portal, click **Compute → Data Centers**.
 - b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
 - c. Click the name of that data center.
 - d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.
 - e. Record the **Domain Name** for use later on.
 - f. Confirm **Free Space** has at least 230 GiB.
 - g. Confirm that the storage domain meets [these etcd backend performance requirements](#), which you [can measure by using the fio performance benchmarking tool](#).
 - h. In the data center details, click the **Clusters** tab.
 - i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.
3. Inspect the RHV host resources.
 - a. In the RHV Administration Portal, click **Compute > Clusters**.
 - b. Click the cluster where you plan to install OpenShift Container Platform.
 - c. In the cluster details, click the **Hosts** tab.
 - d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available *exclusively* for the OpenShift Container Platform cluster.
 - e. Record the number of available **Logical CPU Cores** for use later on.
 - f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.
 - g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:
 - 16 GiB required for the bootstrap machine
 - 16 GiB required for each of the three control plane machines

- 16 GiB for each of the three compute machines
- Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.
- Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager's REST API. From a virtual machine on this network, use curl to reach the RHV Manager's REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1
https://<engine-fqdn>/ovirt-engine/api 2
```

1 For **<username>**, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For **<password>**, specify the password for that user name.

2 For **<engine-fqdn>**, specify the fully qualified domain name of the RHV environment.

For example:

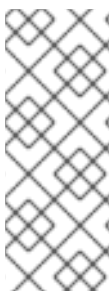
```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

22.4.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by

their fully-qualified domain names in both the node objects and all DNS requests.

Firewall

Configure your firewall so your cluster has access to required sites.

See also:

- [Red Hat Virtualization Manager firewall requirements](#)
- [Host firewall requirements](#)

Load balancers

Configure one or preferably two layer-4 load balancers:

- Provide load balancing for ports **6443** and **22623** on the control plane and bootstrap machines. Port **6443** provides access to the Kubernetes API server and must be reachable both internally and externally. Port **22623** must be accessible to nodes within the cluster.
- Provide load balancing for port **443** and **80** for machines that run the Ingress router, which are usually compute nodes in the default configuration. Both ports must be accessible from within and outside the cluster.

DNS

Configure infrastructure-provided DNS to allow the correct resolution of the main components and services. If you use only one load balancer, these DNS records can point to the same IP address.

- Create DNS records for **api.<cluster_name>.<base_domain>** (internal and external resolution) and **api-int.<cluster_name>.<base_domain>** (internal resolution) that point to the load balancer for the control plane machines.
- Create a DNS record for ***.apps.<cluster_name>.<base_domain>** that points to the load balancer for the Ingress router. For example, ports **443** and **80** of the compute machines.

22.4.5.1. Setting the cluster node hostnames through DHCP

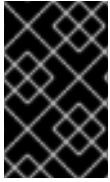
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

22.4.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 22.6. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 22.7. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 22.8. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

22.4.6. Setting up the installation machine

To run the binary **openshift-install** installation program and Ansible scripts, set up the RHV Manager or an Red Hat Enterprise Linux (RHEL) computer with network access to the RHV environment and the REST API on the Manager.

Procedure

1. Update or install Python3 and Ansible. For example:

```
# dnf update python3 ansible
```

2. Install the [python3-ovirt-engine-sdk4](#) package to get the Python Software Development Kit.
3. Install the **ovirt.image-template** Ansible role. On the RHV Manager and other Red Hat Enterprise Linux (RHEL) machines, this role is distributed as the **ovirt-ansible-image-template** package. For example, enter:

```
# dnf install ovirt-ansible-image-template
```

4. Install the **ovirt.vm-infra** Ansible role. On the RHV Manager and other RHEL machines, this role is distributed as the **ovirt-ansible-vm-infra** package.

```
# dnf install ovirt-ansible-vm-infra
```

5. Create an environment variable and assign an absolute or relative path to it. For example, enter:

```
$ export ASSETS_DIR=./wrk
```



NOTE

The installation program uses this variable to create a directory where it saves important installation-related files. Later, the installation process reuses this variable to locate those asset files. Avoid deleting this assets directory; it is required for uninstalling the cluster.

22.4.7. Installing OpenShift Container Platform on RHV in insecure mode

By default, the installer creates a CA certificate, prompts you for confirmation, and stores the certificate to use during installation. You do not need to create or install one manually.

Although it is not recommended, you can override this functionality and install OpenShift Container Platform without verifying a certificate by installing OpenShift Container Platform on RHV in **insecure** mode.



WARNING

Installing in **insecure** mode is not recommended, because it enables a potential attacker to perform a Man-in-the-Middle attack and capture sensitive credentials on the network.

Procedure

1. Create a file named `~/ovirt/ovirt-config.yaml`.
2. Add the following content to **ovirt-config.yaml**:

```
ovirt_url: https://ovirt.example.com/ovirt-engine/api 1
ovirt_fqdn: ovirt.example.com 2
ovirt_pem_url: ""
ovirt_username: ocpadmin@internal
ovirt_password: super-secret-password 3
ovirt_insecure: true
```

- 1 Specify the hostname or address of your oVirt engine.
- 2 Specify the fully qualified domain name of your oVirt engine.
- 3 Specify the admin password for your oVirt engine.

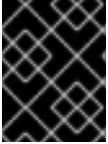
3. Run the installer.

22.4.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

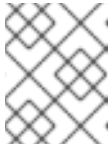
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

22.4.9. Obtaining the installation program

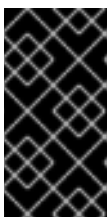
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

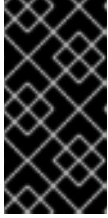
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

22.4.10. Downloading the Ansible playbooks

Download the Ansible playbooks for installing OpenShift Container Platform version 4.12 on RHV.

Procedure

- On your installation machine, run the following commands:

```
$ mkdir playbooks
```

```
$ cd playbooks
```

```
$ xargs -n 1 curl -O <<< '
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/bootstrap.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/common-
  auth.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/create-
  templates-and-vms.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/inventory.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/masters.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  bootstrap.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  masters.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  workers.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/workers.yml'
```

Next steps

- After you download these Ansible playbooks, you must also create the environment variable for the assets directory and customize the **inventory.yml** file before you create an installation configuration file by running the installation program.

22.4.11. The inventory.yml file

You use the **inventory.yml** file to define and create elements of the OpenShift Container Platform cluster you are installing. This includes elements such as the Red Hat Enterprise Linux CoreOS (RHCOS) image, virtual machine templates, bootstrap machine, control plane nodes, and worker nodes. You also use **inventory.yml** to destroy the cluster.

The following **inventory.yml** example shows you the parameters and their default values. The quantities and numbers in these default values meet the requirements for running a production OpenShift Container Platform cluster in a RHV environment.

Example inventory.yml file

```
---
all:
  vars:

    ovirt_cluster: "Default"
    ocp:
      assets_dir: "{{ lookup('env', 'ASSETS_DIR') }}"
      ovirt_config_path: "{{ lookup('env', 'HOME') }}/.ovirt/ovirt-config.yaml"

    # ---
    # {op-system} section
    # ---
    rhcos:
      image_url: "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-
openstack.x86_64.qcow2.gz"
      local_cmp_image_path: "/tmp/rhcos.qcow2.gz"
      local_image_path: "/tmp/rhcos.qcow2"

    # ---
    # Profiles section
    # ---
    control_plane:
      cluster: "{{ ovirt_cluster }}"
      memory: 16GiB
      sockets: 4
      cores: 1
      template: rhcos_tpl
      operating_system: "rhcos_x64"
      type: high_performance
      graphical_console:
        headless_mode: false
      protocol:
        - spice
        - vnc
      disks:
        - size: 120GiB
          name: os
          interface: virtio_scsi
          storage_domain: depot_nvme
      nics:
        - name: nic1
          network: lab
          profile: lab

    compute:
```



```

cluster: "{{ ovirt_cluster }}"
memory: 16GiB
sockets: 4
cores: 1
template: worker_rhcos_tpl
operating_system: "rhcos_x64"
type: high_performance
graphical_console:
  headless_mode: false
protocol:
  - spice
  - vnc
disks:
  - size: 120GiB
    name: os
    interface: virtio_scsi
    storage_domain: depot_nvme
nics:
  - name: nic1
    network: lab
    profile: lab

# ---
# Virtual machines section
# ---
vms:
  - name: "{{ metadata.infraID }}-bootstrap"
    ocp_type: bootstrap
    profile: "{{ control_plane }}"
    type: server
  - name: "{{ metadata.infraID }}-master0"
    ocp_type: master
    profile: "{{ control_plane }}"
  - name: "{{ metadata.infraID }}-master1"
    ocp_type: master
    profile: "{{ control_plane }}"
  - name: "{{ metadata.infraID }}-master2"
    ocp_type: master
    profile: "{{ control_plane }}"
  - name: "{{ metadata.infraID }}-worker0"
    ocp_type: worker
    profile: "{{ compute }}"
  - name: "{{ metadata.infraID }}-worker1"
    ocp_type: worker
    profile: "{{ compute }}"
  - name: "{{ metadata.infraID }}-worker2"
    ocp_type: worker
    profile: "{{ compute }}"

```



IMPORTANT

Enter values for parameters whose descriptions begin with "Enter." Otherwise, you can use the default value or replace it with a new value.

General section

- **ovirt_cluster**: Enter the name of an existing RHV cluster in which to install the OpenShift Container Platform cluster.
- **ocp.assets_dir**: The path of a directory the **openshift-install** installation program creates to store the files that it generates.
- **ocp.ovirt_config_path**: The path of the **ovirt-config.yaml** file the installation program generates, for example, **./wrk/install-config.yaml**. This file contains the credentials required to interact with the REST API of the Manager.

Red Hat Enterprise Linux CoreOS (RHCOS) section

- **image_url**: Enter the URL of the RHCOS image you specified for download.
- **local_cmp_image_path**: The path of a local download directory for the compressed RHCOS image.
- **local_image_path**: The path of a local directory for the extracted RHCOS image.

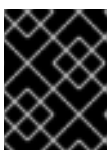
Profiles section

This section consists of two profiles:

- **control_plane**: The profile of the bootstrap and control plane nodes.
- **compute**: The profile of workers nodes in the compute plane.

These profiles have the following parameters. The default values of the parameters meet the minimum requirements for running a production cluster. You can increase or customize these values to meet your workload requirements.

- **cluster**: The value gets the cluster name from **ovirt_cluster** in the General Section.
- **memory**: The amount of memory, in GB, for the virtual machine.
- **sockets**: The number of sockets for the virtual machine.
- **cores**: The number of cores for the virtual machine.
- **template**: The name of the virtual machine template. If plan to install multiple clusters, and these clusters use templates that contain different specifications, prepend the template name with the ID of the cluster.
- **operating_system**: The type of guest operating system in the virtual machine. With oVirt/RHV version 4.4, this value must be **rhcos_x64** so the value of **ignition script** can be passed to the VM.
- **type**: Enter **server** as the type of the virtual machine.



IMPORTANT

You must change the value of the **type** parameter from **high_performance** to **server**.

- **disks**: The disk specifications. The **control_plane** and **compute** nodes can have different storage domains.

- **size**: The minimum disk size.
- **name**: Enter the name of a disk connected to the target cluster in RHV.
- **interface**: Enter the interface type of the disk you specified.
- **storage_domain**: Enter the storage domain of the disk you specified.
- **nics**: Enter the **name** and **network** the virtual machines use. You can also specify the virtual network interface profile. By default, NICs obtain their MAC addresses from the oVirt/RHV MAC pool.

Virtual machines section

This final section, **vms**, defines the virtual machines you plan to create and deploy in the cluster. By default, it provides the minimum number of control plane and worker nodes for a production environment.

vms contains three required elements:

- **name**: The name of the virtual machine. In this case, **metadata.infraID** prepends the virtual machine name with the infrastructure ID from the **metadata.yml** file.
- **ocp_type**: The role of the virtual machine in the OpenShift Container Platform cluster. Possible values are **bootstrap**, **master**, **worker**.
- **profile**: The name of the profile from which each virtual machine inherits specifications. Possible values in this example are **control_plane** or **compute**.
You can override the value a virtual machine inherits from its profile. To do this, you add the name of the profile attribute to the virtual machine in **inventory.yml** and assign it an overriding value. To see an example of this, examine the **name: "{{ metadata.infraID }}-bootstrap"** virtual machine in the preceding **inventory.yml** example: It has a **type** attribute whose value, **server**, overrides the value of the **type** attribute this virtual machine would otherwise inherit from the **control_plane** profile.

Metadata variables

For virtual machines, **metadata.infraID** prepends the name of the virtual machine with the infrastructure ID from the **metadata.json** file you create when you build the Ignition files.

The playbooks use the following code to read **infraID** from the specific file located in the **ocp.assets_dir**.

```
---
- name: include metadata.json vars
  include_vars:
    file: "{{ ocp.assets_dir }}/metadata.json"
    name: metadata
...
```

22.4.12. Specifying the RHCOS image settings

Update the Red Hat Enterprise Linux CoreOS (RHCOS) image settings of the **inventory.yml** file. Later, when you run this file one of the playbooks, it downloads a compressed Red Hat Enterprise Linux CoreOS (RHCOS) image from the **image_url** URL to the **local_cmp_image_path** directory. The

playbook then uncompresses the image to the **local_image_path** directory and uses it to create oVirt/RHV templates.

Procedure

1. Locate the RHCOS image download page for the version of OpenShift Container Platform you are installing, such as [Index of /pub/openshift-v4/dependencies/rhcos/latest/latest](#).
2. From that download page, copy the URL of an OpenStack **qcow2** image, such as **https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-openshift.x86_64.qcow2.gz**.
3. Edit the **inventory.yml** playbook you downloaded earlier. In it, paste the URL as the value for **image_url**. For example:

```
rhcos:
  "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-openshift.x86_64.qcow2.gz"
```

22.4.13. Creating the install config file

You create an installation configuration file by running the installation program, **openshift-install**, and responding to its prompts with information you specified or gathered earlier.

When you finish responding to the prompts, the installation program creates an initial version of the **install-config.yaml** file in the assets directory you specified earlier, for example, **./wrk/install-config.yaml**

The installation program also creates a file, **\$HOME/.ovirt/ovirt-config.yaml**, that contains all the connection parameters that are required to reach the Manager and use its REST API.

NOTE: The installation process does not use values you supply for some parameters, such as **Internal API virtual IP** and **Ingress virtual IP**, because you have already configured them in your infrastructure DNS.

It also uses the values you supply for parameters in **inventory.yml**, like the ones for **oVirt cluster**, **oVirt storage**, and **oVirt network**. And uses a script to remove or replace these same values from **install-config.yaml** with the previously mentioned **virtual IPs**.

Procedure

1. Run the installation program:

```
$ openshift-install create install-config --dir $ASSETS_DIR
```

2. Respond to the installation program's prompts with information about your system.

Example output

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <*****>
? oVirt cluster <cluster>
```

```
? oVirt storage <storage>
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <*****>
```

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <*****>
? oVirt cluster <cluster>
? oVirt storage <storage>
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <*****>
```

For **Internal API virtual IP** and **Ingress virtual IP**, supply the IP addresses you specified when you configured the DNS service.

Together, the values you enter for the **oVirt cluster** and **Base Domain** prompts form the FQDN portion of URLs for the REST API and any applications you create, such as **https://api.ocp4.example.org:6443/** and **https://console-openshift-console.apps.ocp4.example.org**.

You can get the [pull secret from the Red Hat OpenShift Cluster Manager](#) .

22.4.14. Customizing install-config.yaml

Here, you use three Python scripts to override some of the installation program's default behaviors:

- By default, the installation program uses the machine API to create nodes. To override this default behavior, you set the number of compute nodes to zero replicas. Later, you use Ansible playbooks to create the compute nodes.
- By default, the installation program sets the IP range of the machine network for nodes. To override this default behavior, you set the IP range to match your infrastructure.
- By default, the installation program sets the platform to **ovirt**. However, installing a cluster on user-provisioned infrastructure is more similar to installing a cluster on bare metal. Therefore, you delete the ovirt platform section from **install-config.yaml** and change the platform to **none**. Instead, you use **inventory.yml** to specify all of the required settings.



NOTE

These snippets work with Python 3 and Python 2.

Procedure

1. Set the number of compute nodes to zero replicas:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
conf["compute"][0]["replicas"] = 0
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

2. Set the IP range of the machine network. For example, to set the range to **172.16.0.0/16**, enter:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
conf["networking"]["machineNetwork"][0]["cidr"] = "172.16.0.0/16"
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

3. Remove the **ovirt** section and change the platform to **none**:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
platform = conf["platform"]
del platform["ovirt"]
platform["none"] = {}
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```



WARNING

Red Hat Virtualization does not currently support installation with user-provisioned infrastructure on the oVirt platform. Therefore, you must set the platform to **none**, allowing OpenShift Container Platform to identify each node as a bare-metal node and the cluster as a bare-metal cluster. This is the same as [installing a cluster on any platform](#), and has the following limitations:

1. There will be no cluster provider so you must manually add each machine and there will be no node scaling capabilities.
2. The oVirt CSI driver will not be installed and there will be no CSI capabilities.

22.4.15. Generate manifest files

Use the installation program to generate a set of manifest files in the assets directory.

The command to generate the manifest files displays a warning message before it consumes the **install-config.yaml** file.

If you plan to reuse the **install-config.yaml** file, create a backup copy of it before you back it up before you generate the manifest files.

Procedure

- Optional: Create a backup copy of the **install-config.yaml** file:

```
$ cp install-config.yaml install-config.yaml.backup
```

- Generate a set of manifests in your assets directory:

```
$ openshift-install create manifests --dir $ASSETS_DIR
```

This command displays the following messages.

Example output

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

The command generates the following manifest files:

Example output

```
$ tree
.
├── wrk
│   ├── manifests
│   │   ├── 04-openshift-machine-config-operator.yaml
│   │   ├── cluster-config.yaml
│   │   ├── cluster-dns-02-config.yml
│   │   ├── cluster-infrastructure-02-config.yml
│   │   ├── cluster-ingress-02-config.yml
│   │   ├── cluster-network-01-crd.yml
│   │   ├── cluster-network-02-config.yml
│   │   ├── cluster-proxy-01-config.yaml
│   │   ├── cluster-scheduler-02-config.yml
│   │   ├── cvo-overrides.yaml
│   │   ├── etcd-ca-bundle-configmap.yaml
│   │   ├── etcd-client-secret.yaml
│   │   ├── etcd-host-service-endpoints.yaml
│   │   ├── etcd-host-service.yaml
│   │   ├── etcd-metric-client-secret.yaml
│   │   ├── etcd-metric-serving-ca-configmap.yaml
│   │   ├── etcd-metric-signer-secret.yaml
│   │   ├── etcd-namespace.yaml
│   │   ├── etcd-service.yaml
│   │   ├── etcd-serving-ca-configmap.yaml
│   │   ├── etcd-signer-secret.yaml
│   │   ├── kube-cloud-config.yaml
│   │   ├── kube-system-configmap-root-ca.yaml
│   │   ├── machine-config-server-tls-secret.yaml
│   │   └── openshift-config-secret-pull-secret.yaml
│   └── openshift
│       ├── 99_kubeadmin-password-secret.yaml
│       ├── 99_openshift-cluster-api_master-user-data-secret.yaml
│       └── 99_openshift-cluster-api_worker-user-data-secret.yaml
```

```

├── 99_openshift-machineconfig_99-master-ssh.yaml
├── 99_openshift-machineconfig_99-worker-ssh.yaml
└── openshift-install-manifests.yaml

```

Next steps

- Make control plane nodes non-schedulable.

22.4.16. Making control-plane nodes non-schedulable

Because you are manually creating and deploying the control plane machines, you must configure a manifest file to make the control plane nodes non-schedulable.

Procedure

1. To make the control plane nodes non-schedulable, enter:

```

$ python3 -c 'import os, yaml
path = "%s/manifests/cluster-scheduler-02-config.yml" % os.environ["ASSETS_DIR"]
data = yaml.safe_load(open(path))
data["spec"]["mastersSchedulable"] = False
open(path, "w").write(yaml.dump(data, default_flow_style=False))'

```

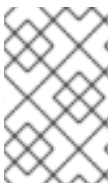
22.4.17. Building the Ignition files

To build the Ignition files from the manifest files you just generated and modified, you run the installation program. This action creates a Red Hat Enterprise Linux CoreOS (RHCOS) machine, **inframfs**, which fetches the Ignition files and performs the configurations needed to create a node.

In addition to the Ignition files, the installation program generates the following:

- An **auth** directory that contains the admin credentials for connecting to the cluster with the **oc** and **kubectl** utilities.
- A **metadata.json** file that contains information such as the OpenShift Container Platform cluster name, cluster ID, and infrastructure ID for the current installation.

The Ansible playbooks for this installation process use the value of **infrald** as a prefix for the virtual machines they create. This prevents naming conflicts when there are multiple installations in the same oVirt/RHV cluster.



NOTE

Certificates in Ignition configuration files expire after 24 hours. Complete the cluster installation and keep the cluster running in a non-degraded state for 24 hours so that the first certificate rotation can finish.

Procedure

1. To build the Ignition files, enter:

```

$ openshift-install create ignition-configs --dir $ASSETS_DIR

```


Example output

```
$ tree
.
├── wrk
│   ├── auth
│   │   ├── kubeadmin-password
│   │   └── kubeconfig
│   ├── bootstrap.ign
│   ├── master.ign
│   ├── metadata.json
│   └── worker.ign
```

22.4.18. Creating templates and virtual machines

After confirming the variables in the **inventory.yml**, you run the first Ansible provisioning playbook, **create-templates-and-vms.yml**.

This playbook uses the connection parameters for the RHV Manager from **\$HOME/.ovirt/ovirt-config.yaml** and reads **metadata.json** in the assets directory.

If a local Red Hat Enterprise Linux CoreOS (RHCOS) image is not already present, the playbook downloads one from the URL you specified for **image_url** in **inventory.yml**. It extracts the image and uploads it to RHV to create templates.

The playbook creates a template based on the **control_plane** and **compute** profiles in the **inventory.yml** file. If these profiles have different names, it creates two templates.

When the playbook finishes, the virtual machines it creates are stopped. You can get information from them to help configure other infrastructure elements. For example, you can get the virtual machines' MAC addresses to configure DHCP to assign permanent IP addresses to the virtual machines.

Procedure

1. In **inventory.yml**, under the **control_plane** and **compute** variables, change both instances of **type: high_performance** to **type: server**.
2. Optional: If you plan to perform multiple installations to the same cluster, create different templates for each OpenShift Container Platform installation. In the **inventory.yml** file, prepend the value of **template** with **infraID**. For example:

```
control_plane:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
  template: "{{ metadata.infraID }}-rhcos_tpl"
  operating_system: "rhcos_x64"
  ...
```

3. Create the templates and virtual machines:

```
$ ansible-playbook -i inventory.yml create-templates-and-vms.yml
```

22.4.19. Creating the bootstrap machine

You create a bootstrap machine by running the **bootstrap.yml** playbook. This playbook starts the bootstrap virtual machine, and passes it the **bootstrap.ign** Ignition file from the assets directory. The bootstrap node configures itself so it can serve Ignition files to the control plane nodes.

To monitor the bootstrap process, you use the console in the RHV Administration Portal or connect to the virtual machine by using SSH.

Procedure

1. Create the bootstrap machine:

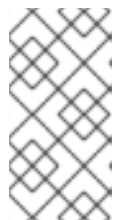
```
$ ansible-playbook -i inventory.yml bootstrap.yml
```

2. Connect to the bootstrap machine using a console in the Administration Portal or SSH. Replace **<bootstrap_ip>** with the bootstrap node IP address. To use SSH, enter:

```
$ ssh core@<bootstrap.ip>
```

3. Collect **bootkube.service** journald unit logs for the release image service from the bootstrap node:

```
[core@ocp4-1k6b4-bootstrap ~]$ journalctl -b -f -u release-image.service -u bootkube.service
```



NOTE

The **bootkube.service** log on the bootstrap node outputs etcd **connection refused** errors, indicating that the bootstrap server is unable to connect to etcd on control plane nodes. After etcd has started on each control plane node and the nodes have joined the cluster, the errors should stop.

22.4.20. Creating the control plane nodes

You create the control plane nodes by running the **masters.yml** playbook. This playbook passes the **master.ign** Ignition file to each of the virtual machines. The Ignition file contains a directive for the control plane node to get the Ignition from a URL such as <https://api-int.ocp4.example.org:22623/config/master>. The port number in this URL is managed by the load balancer, and is accessible only inside the cluster.

Procedure

1. Create the control plane nodes:

```
$ ansible-playbook -i inventory.yml masters.yml
```

2. While the playbook creates your control plane, monitor the bootstrapping process:

```
$ openshift-install wait-for bootstrap-complete --dir $ASSETS_DIR
```

Example output

```
INFO API v1.25.0 up
INFO Waiting up to 40m0s for bootstrapping to complete...
```

- When all the pods on the control plane nodes and etcd are up and running, the installation program displays the following output.

Example output

```
INFO It is now safe to remove the bootstrap resources
```

22.4.21. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

- In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=$ASSETS_DIR/auth/kubeconfig
```

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

- View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

- View your cluster's version:

```
$ oc get clusterversion
```

- View your Operators' status:

```
$ oc get clusteroperator
```

- View all running pods in the cluster:

```
$ oc get pods -A
```

22.4.22. Removing the bootstrap machine

After the **wait-for** command shows that the bootstrap process is complete, you must remove the bootstrap virtual machine to free up compute, memory, and storage resources. Also, remove settings for the bootstrap machine from the load balancer directives.

Procedure

- To remove the bootstrap machine from the cluster, enter:

```
$ ansible-playbook -i inventory.yml retire-bootstrap.yml
```

2. Remove settings for the bootstrap machine from the load balancer directives.

22.4.23. Creating the worker nodes and completing the installation

Creating worker nodes is similar to creating control plane nodes. However, worker nodes workers do not automatically join the cluster. To add them to the cluster, you review and approve the workers' pending CSRs (Certificate Signing Requests).

After approving the first requests, you continue approving CSR until all of the worker nodes are approved. When you complete this process, the worker nodes become **Ready** and can have pods scheduled to run on them.

Finally, monitor the command line to see when the installation process completes.

Procedure

1. Create the worker nodes:

```
$ ansible-playbook -i inventory.yml workers.yml
```

2. To list all of the CSRs, enter:

```
$ oc get csr -A
```

Eventually, this command displays one CSR per node. For example:

Example output

```
NAME          AGE  SIGNERNAME                                REQUESTOR
CONDITION
csr-2lnxd    63m  kubernetes.io/kubelet-serving            system:node:ocp4-lk6b4-
master0.ocp4.example.org                Approved,Issued
csr-hff4q    64m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-hsn96    60m  kubernetes.io/kubelet-serving            system:node:ocp4-lk6b4-
master2.ocp4.example.org                Approved,Issued
csr-m724n    6m2s kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
csr-p4dz2    60m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-t9vfj    60m  kubernetes.io/kubelet-serving            system:node:ocp4-lk6b4-
master1.ocp4.example.org                Approved,Issued
csr-tggtr    61m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-wcbrf    7m6s kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
```

3. To filter the list and see only pending CSRs, enter:

```
$ watch "oc get csr -A | grep pending -i"
```

This command refreshes the output every two seconds and displays only pending CSRs. For example:

Example output

```
Every 2.0s: oc get csr -A | grep pending -i
csr-m724n 10m kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
csr-wcbrf 11m kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
```

4. Inspect each pending request. For example:

Example output

```
$ oc describe csr csr-m724n
```

Example output

```
Name:          csr-m724n
Labels:        <none>
Annotations:   <none>
CreationTimestamp: Sun, 19 Jul 2020 15:59:37 +0200
Requesting User: system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper
Signer:        kubernetes.io/kube-apiserver-client-kubelet
Status:        Pending
Subject:
  Common Name:  system:node:ocp4-1k6b4-worker1.ocp4.example.org
  Serial Number:
  Organization: system:nodes
Events: <none>
```

5. If the CSR information is correct, approve the request:

```
$ oc adm certificate approve csr-m724n
```

6. Wait for the installation process to finish:

```
$ openshift-install wait-for install-complete --dir $ASSETS_DIR --log-level debug
```

When the installation completes, the command line displays the URL of the OpenShift Container Platform web console and the administrator user name and password.

22.4.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct,

either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

22.5. INSTALLING A CLUSTER ON RHV IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a customized OpenShift Container Platform cluster on Red Hat Virtualization (RHV) in a restricted network by creating an internal mirror of the installation release content.

22.5.1. Prerequisites

The following items are required to install an OpenShift Container Platform cluster on a RHV environment.

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You have a supported combination of versions in the [Support Matrix for OpenShift Container Platform on RHV](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

22.5.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to

its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

22.5.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

22.5.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

22.5.4. Requirements for the RHV environment

To install and run an OpenShift Container Platform version 4.12 cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on **default** values multiplied by the default number of virtual machines the installation program creates. These resources must be available **in addition to** what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

Requirements

- The RHV version is 4.4.
- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
 - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
 - 112 GiB RAM or more, including:
 - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
 - 16 GiB or more for each of the three control plane machines which provide the control plane.
 - 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet [these etcd backend performance requirements](#).
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
 - **DiskOperator**
 - **DiskCreator**
 - **UserTemplateBasedVm**
 - **TemplateOwner**

- **TemplateCreator**
- **ClusterAdmin** on the target cluster



WARNING

Apply the principle of least privilege: Avoid using an administrator account with **SuperUser** privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary **ovirt-config.yaml** file that might be compromised.

22.5.5. Verifying the requirements for the RHV environment

Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.



IMPORTANT

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check that the RHV version supports installation of OpenShift Container Platform version 4.12.
 - a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select **About**.
 - b. In the window that opens, make a note of the **RHV Software Version**
 - c. Confirm that the RHV version is 4.4. For more information about supported version combinations, see [Support Matrix for OpenShift Container Platform on RHV](#).
2. Inspect the data center, cluster, and storage.
 - a. In the RHV Administration Portal, click **Compute → Data Centers**.
 - b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
 - c. Click the name of that data center.
 - d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.
 - e. Record the **Domain Name** for use later on.
 - f. Confirm **Free Space** has at least 230 GiB.

- g. Confirm that the storage domain meets [these etcd backend performance requirements](#), which you [can measure by using the fio performance benchmarking tool](#).
 - h. In the data center details, click the **Clusters** tab.
 - i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.
3. Inspect the RHV host resources.
 - a. In the RHV Administration Portal, click **Compute > Clusters**.
 - b. Click the cluster where you plan to install OpenShift Container Platform.
 - c. In the cluster details, click the **Hosts** tab.
 - d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available *exclusively* for the OpenShift Container Platform cluster.
 - e. Record the number of available **Logical CPU Cores** for use later on.
 - f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.
 - g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:
 - 16 GiB required for the bootstrap machine
 - 16 GiB required for each of the three control plane machines
 - 16 GiB for each of the three compute machines
 - h. Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.
 4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager's REST API. From a virtual machine on this network, use curl to reach the RHV Manager's REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 1  
https://<engine-fqdn>/ovirt-engine/api 2
```

- 1 For **<username>**, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For **<password>**, specify the password for that user name.

- 2 For **<engine-fqdn>**, specify the fully qualified domain name of the RHV environment.

For example:

```
$ curl -k -u ocpadmin@internal:pw123 \  
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

22.5.6. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

Firewall

Configure your firewall so your cluster has access to required sites.

See also:

- [Red Hat Virtualization Manager firewall requirements](#)
- [Host firewall requirements](#)

DNS

Configure infrastructure-provided DNS to allow the correct resolution of the main components and services. If you use only one load balancer, these DNS records can point to the same IP address.

- Create DNS records for **api.<cluster_name>.<base_domain>** (internal and external resolution) and **api-int.<cluster_name>.<base_domain>** (internal resolution) that point to the load balancer for the control plane machines.
- Create a DNS record for ***.apps.<cluster_name>.<base_domain>** that points to the load balancer for the Ingress router. For example, ports **443** and **80** of the compute machines.

22.5.6.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not

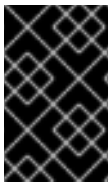
provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

22.5.6.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 22.9. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 22.10. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 22.11. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

22.5.7. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are

provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 22.12. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>
Routes	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
		For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

Component	Record	Description
Control plane machines	<control_plane><n>. <cluster_name>. <base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>. <cluster_name>. <base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**NOTE**

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

22.5.7.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 22.1. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
```

```

helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 22.2. Sample DNS zone database for reverse records

```

$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial

```



```

3H ; refresh (3 hours)
30M ; retry (30 minutes)
2W ; expiry (2 weeks)
1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF

```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

22.5.7.2. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



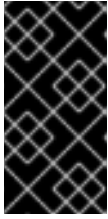
NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 22.13. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 22.14. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

22.5.7.2.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 22.3. Sample API and application Ingress load balancer configuration

```
global
log      127.0.0.1 local2
pidfile  /var/run/haproxy.pid
maxconn  4000
daemon
defaults
```

```

mode          http
log           global
option       dontlognull
option http-server-close
option       redispatch
retries      3
timeout http-request 10s
timeout queue 1m
timeout connect 10s
timeout client 1m
timeout server 1m
timeout http-keep-alive 10s
timeout check 10s
maxconn      3000
listen api-server-6443 1
bind *:6443
mode tcp
option httpchk GET /readyz HTTP/1.0
option log-health-checks
balance roundrobin
server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup 2
server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.

- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

22.5.8. Setting up the installation machine

To run the binary **openshift-install** installation program and Ansible scripts, set up the RHV Manager or an Red Hat Enterprise Linux (RHEL) computer with network access to the RHV environment and the REST API on the Manager.

Procedure

1. Update or install Python3 and Ansible. For example:

```
# dnf update python3 ansible
```

2. Install the [python3-ovirt-engine-sdk4](#) package to get the Python Software Development Kit.
3. Install the **ovirt.image-template** Ansible role. On the RHV Manager and other Red Hat Enterprise Linux (RHEL) machines, this role is distributed as the **ovirt-ansible-image-template** package. For example, enter:

```
# dnf install ovirt-ansible-image-template
```

4. Install the **ovirt.vm-infra** Ansible role. On the RHV Manager and other RHEL machines, this role is distributed as the **ovirt-ansible-vm-infra** package.

```
# dnf install ovirt-ansible-vm-infra
```

5. Create an environment variable and assign an absolute or relative path to it. For example, enter:

```
$ export ASSETS_DIR=./wrk
```

**NOTE**

The installation program uses this variable to create a directory where it saves important installation-related files. Later, the installation process reuses this variable to locate those asset files. Avoid deleting this assets directory; it is required for uninstalling the cluster.

22.5.9. Setting up the CA certificate for RHV

Download the CA certificate from the Red Hat Virtualization (RHV) Manager and set it up on the installation machine.

You can download the certificate from a webpage on the RHV Manager or by using a **curl** command.

Later, you provide the certificate to the installation program.

Procedure

1. Use either of these two methods to download the CA certificate:

- Go to the Manager's webpage, **<https://<engine-fqdn>/ovirt-engine/>**. Then, under **Downloads**, click the **CA Certificate** link.
- Run the following command:

```
$ curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem 1
```

- 1** For **<engine-fqdn>**, specify the fully qualified domain name of the RHV Manager, such as **rhv-env.virtlab.example.com**.

2. Configure the CA file to grant rootless user access to the Manager. Set the CA file permissions to have an octal value of **0644** (symbolic value: **-rw-r--r--**):

```
$ sudo chmod 0644 /tmp/ca.pem
```

3. For Linux, copy the CA certificate to the directory for server certificates. Use **-p** to preserve the permissions:

```
$ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
```

4. Add the certificate to the certificate manager for your operating system:

- For macOS, double-click the certificate file and use the **Keychain Access** utility to add the file to the **System** keychain.
- For Linux, update the CA trust:

```
$ sudo update-ca-trust
```

**NOTE**

If you use your own certificate authority, make sure the system trusts it.

Additional resources

- To learn more, see [Authentication and Security](#) in the RHV documentation.

22.5.10. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

22.5.11. Downloading the Ansible playbooks

Download the Ansible playbooks for installing OpenShift Container Platform version 4.12 on RHV.

Procedure

- On your installation machine, run the following commands:

```
$ mkdir playbooks
```

```
▪
```



```

$ cd playbooks

$ xargs -n 1 curl -O <<< '
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/bootstrap.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/common-
  auth.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/create-
  templates-and-vms.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/inventory.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/masters.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  bootstrap.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  masters.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/retire-
  workers.yml
  https://raw.githubusercontent.com/openshift/installer/release-4.12/upi/ovirt/workers.yml'

```

Next steps

- After you download these Ansible playbooks, you must also create the environment variable for the assets directory and customize the **inventory.yml** file before you create an installation configuration file by running the installation program.

22.5.12. The inventory.yml file

You use the **inventory.yml** file to define and create elements of the OpenShift Container Platform cluster you are installing. This includes elements such as the Red Hat Enterprise Linux CoreOS (RHCOS) image, virtual machine templates, bootstrap machine, control plane nodes, and worker nodes. You also use **inventory.yml** to destroy the cluster.

The following **inventory.yml** example shows you the parameters and their default values. The quantities and numbers in these default values meet the requirements for running a production OpenShift Container Platform cluster in a RHV environment.

Example inventory.yml file

```

---
all:
  vars:

    ovirt_cluster: "Default"
    ocp:
      assets_dir: "{{ lookup('env', 'ASSETS_DIR') }}"
      ovirt_config_path: "{{ lookup('env', 'HOME') }}/ovirt/ovirt-config.yaml"

    # ---
    # {op-system} section
    # ---
    rhcos:
      image_url: "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-
      openstack.x86_64.qcow2.gz"
      local_cmp_image_path: "/tmp/rhcos.qcow2.gz"
      local_image_path: "/tmp/rhcos.qcow2"

```

```
# ---
# Profiles section
# ---
control_plane:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
  template: rhcos_tpl
  operating_system: "rhcos_x64"
  type: high_performance
  graphical_console:
    headless_mode: false
  protocol:
    - spice
    - vnc
  disks:
    - size: 120GiB
      name: os
      interface: virtio_scsi
      storage_domain: depot_nvme
  nics:
    - name: nic1
      network: lab
      profile: lab

compute:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
  template: worker_rhcos_tpl
  operating_system: "rhcos_x64"
  type: high_performance
  graphical_console:
    headless_mode: false
  protocol:
    - spice
    - vnc
  disks:
    - size: 120GiB
      name: os
      interface: virtio_scsi
      storage_domain: depot_nvme
  nics:
    - name: nic1
      network: lab
      profile: lab

# ---
# Virtual machines section
# ---
vms:
  - name: "{{ metadata.infraID }}-bootstrap"
    ocp_type: bootstrap
```

```

profile: "{{ control_plane }}"
type: server
- name: "{{ metadata.infraID }}-master0"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-master1"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-master2"
  ocp_type: master
  profile: "{{ control_plane }}"
- name: "{{ metadata.infraID }}-worker0"
  ocp_type: worker
  profile: "{{ compute }}"
- name: "{{ metadata.infraID }}-worker1"
  ocp_type: worker
  profile: "{{ compute }}"
- name: "{{ metadata.infraID }}-worker2"
  ocp_type: worker
  profile: "{{ compute }}"

```



IMPORTANT

Enter values for parameters whose descriptions begin with "Enter." Otherwise, you can use the default value or replace it with a new value.

General section

- **ovirt_cluster**: Enter the name of an existing RHV cluster in which to install the OpenShift Container Platform cluster.
- **ocp.assets_dir**: The path of a directory the **openshift-install** installation program creates to store the files that it generates.
- **ocp.ovirt_config_path**: The path of the **ovirt-config.yaml** file the installation program generates, for example, **./wrk/install-config.yaml**. This file contains the credentials required to interact with the REST API of the Manager.

Red Hat Enterprise Linux CoreOS (RHCOS) section

- **image_url**: Enter the URL of the RHCOS image you specified for download.
- **local_cmp_image_path**: The path of a local download directory for the compressed RHCOS image.
- **local_image_path**: The path of a local directory for the extracted RHCOS image.

Profiles section

This section consists of two profiles:

- **control_plane**: The profile of the bootstrap and control plane nodes.
- **compute**: The profile of workers nodes in the compute plane.

These profiles have the following parameters. The default values of the parameters meet the minimum requirements for running a production cluster. You can increase or customize these values to meet your workload requirements.

- **cluster:** The value gets the cluster name from **ovirt_cluster** in the General Section.
- **memory:** The amount of memory, in GB, for the virtual machine.
- **sockets:** The number of sockets for the virtual machine.
- **cores:** The number of cores for the virtual machine.
- **template:** The name of the virtual machine template. If plan to install multiple clusters, and these clusters use templates that contain different specifications, prepend the template name with the ID of the cluster.
- **operating_system:** The type of guest operating system in the virtual machine. With oVirt/RHV version 4.4, this value must be **rhcos_x64** so the value of **Ignition script** can be passed to the VM.
- **type:** Enter **server** as the type of the virtual machine.



IMPORTANT

You must change the value of the **type** parameter from **high_performance** to **server**.

- **disks:** The disk specifications. The **control_plane** and **compute** nodes can have different storage domains.
- **size:** The minimum disk size.
- **name:** Enter the name of a disk connected to the target cluster in RHV.
- **interface:** Enter the interface type of the disk you specified.
- **storage_domain:** Enter the storage domain of the disk you specified.
- **nics:** Enter the **name** and **network** the virtual machines use. You can also specify the virtual network interface profile. By default, NICs obtain their MAC addresses from the oVirt/RHV MAC pool.

Virtual machines section

This final section, **vms**, defines the virtual machines you plan to create and deploy in the cluster. By default, it provides the minimum number of control plane and worker nodes for a production environment.

vms contains three required elements:

- **name:** The name of the virtual machine. In this case, **metadata.infraID** prepends the virtual machine name with the infrastructure ID from the **metadata.yml** file.
- **ocp_type:** The role of the virtual machine in the OpenShift Container Platform cluster. Possible values are **bootstrap**, **master**, **worker**.

- **profile**: The name of the profile from which each virtual machine inherits specifications. Possible values in this example are **control_plane** or **compute**.
You can override the value a virtual machine inherits from its profile. To do this, you add the name of the profile attribute to the virtual machine in **inventory.yml** and assign it an overriding value. To see an example of this, examine the **name: "{{ metadata.infraID }}-bootstrap"** virtual machine in the preceding **inventory.yml** example: It has a **type** attribute whose value, **server**, overrides the value of the **type** attribute this virtual machine would otherwise inherit from the **control_plane** profile.

Metadata variables

For virtual machines, **metadata.infraID** prepends the name of the virtual machine with the infrastructure ID from the **metadata.json** file you create when you build the Ignition files.

The playbooks use the following code to read **infraID** from the specific file located in the **ocp.assets_dir**.

```
---
- name: include metadata.json vars
  include_vars:
    file: "{{ ocp.assets_dir }}/metadata.json"
    name: metadata
...
```

22.5.13. Specifying the RHCOS image settings

Update the Red Hat Enterprise Linux CoreOS (RHCOS) image settings of the **inventory.yml** file. Later, when you run this file one of the playbooks, it downloads a compressed Red Hat Enterprise Linux CoreOS (RHCOS) image from the **image_url** URL to the **local_cmp_image_path** directory. The playbook then uncompresses the image to the **local_image_path** directory and uses it to create oVirt/RHV templates.

Procedure

1. Locate the RHCOS image download page for the version of OpenShift Container Platform you are installing, such as [Index of /pub/openshift-v4/dependencies/rhcos/latest/latest](#).
2. From that download page, copy the URL of an OpenStack **qcow2** image, such as **https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-openstack.x86_64.qcow2.gz**.
3. Edit the **inventory.yml** playbook you downloaded earlier. In it, paste the URL as the value for **image_url**. For example:

```
rhcos:
  "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.12/latest/rhcos-
  openstack.x86_64.qcow2.gz"
```

22.5.14. Creating the install config file

You create an installation configuration file by running the installation program, **openshift-install**, and responding to its prompts with information you specified or gathered earlier.

When you finish responding to the prompts, the installation program creates an initial version of the **install-config.yaml** file in the assets directory you specified earlier, for example, **./wrk/install-config.yaml**

The installation program also creates a file, **\$HOME/.ovirt/ovirt-config.yaml**, that contains all the connection parameters that are required to reach the Manager and use its REST API.

NOTE: The installation process does not use values you supply for some parameters, such as **Internal API virtual IP** and **Ingress virtual IP**, because you have already configured them in your infrastructure DNS.

It also uses the values you supply for parameters in **inventory.yml**, like the ones for **oVirt cluster**, **oVirt storage**, and **oVirt network**. And uses a script to remove or replace these same values from **install-config.yaml** with the previously mentioned **virtual IPs**.

Procedure

1. Run the installation program:

```
$ openshift-install create install-config --dir $ASSETS_DIR
```

2. Respond to the installation program's prompts with information about your system.

Example output

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <*****>
? oVirt cluster <cluster>
? oVirt storage <storage>
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <*****>
```

```
? SSH Public Key /home/user/.ssh/id_dsa.pub
? Platform <ovirt>
? Engine FQDN[:PORT] [? for help] <engine.fqdn>
? Enter ovirt-engine username <ocpadmin@internal>
? Enter password <*****>
? oVirt cluster <cluster>
? oVirt storage <storage>
? oVirt network <net>
? Internal API virtual IP <172.16.0.252>
? Ingress virtual IP <172.16.0.251>
? Base Domain <example.org>
? Cluster Name <ocp4>
? Pull Secret [? for help] <*****>
```

For **Internal API virtual IP** and **Ingress virtual IP**, supply the IP addresses you specified when you configured the DNS service.

Together, the values you enter for the **oVirt cluster** and **Base Domain** prompts form the FQDN portion of URLs for the REST API and any applications you create, such as <https://api.ocp4.example.org:6443/> and <https://console-openshift-console.apps.ocp4.example.org>.

You can get the [pull secret from the Red Hat OpenShift Cluster Manager](#).

22.5.15. Sample install-config.yaml file for RHV

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.

**IMPORTANT**

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. In user-provisioned installations, you must manually deploy the compute machines before you finish installing the cluster.

**NOTE**

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

**NOTE**

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for RHV infrastructure.



IMPORTANT

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

14

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

15

The [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

16

The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

22.5.15.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

22.5.16. Customizing install-config.yaml

Here, you use three Python scripts to override some of the installation program's default behaviors:

- By default, the installation program uses the machine API to create nodes. To override this default behavior, you set the number of compute nodes to zero replicas. Later, you use Ansible playbooks to create the compute nodes.
- By default, the installation program sets the IP range of the machine network for nodes. To override this default behavior, you set the IP range to match your infrastructure.
- By default, the installation program sets the platform to **ovirt**. However, installing a cluster on user-provisioned infrastructure is more similar to installing a cluster on bare metal. Therefore, you delete the ovirt platform section from **install-config.yaml** and change the platform to **none**. Instead, you use **inventory.yml** to specify all of the required settings.

**NOTE**

These snippets work with Python 3 and Python 2.

Procedure

1. Set the number of compute nodes to zero replicas:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
conf["compute"][0]["replicas"] = 0
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

2. Set the IP range of the machine network. For example, to set the range to **172.16.0.0/16**, enter:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))'
```

```
conf["networking"]["machineNetwork"][0]["cidr"] = "172.16.0.0/16"
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```

3. Remove the **ovirt** section and change the platform to **none**:

```
$ python3 -c 'import os, yaml
path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
conf = yaml.safe_load(open(path))
platform = conf["platform"]
del platform["ovirt"]
platform["none"] = {}
open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
```



WARNING

Red Hat Virtualization does not currently support installation with user-provisioned infrastructure on the oVirt platform. Therefore, you must set the platform to **none**, allowing OpenShift Container Platform to identify each node as a bare-metal node and the cluster as a bare-metal cluster. This is the same as [installing a cluster on any platform](#), and has the following limitations:

1. There will be no cluster provider so you must manually add each machine and there will be no node scaling capabilities.
2. The oVirt CSI driver will not be installed and there will be no CSI capabilities.

22.5.17. Generate manifest files

Use the installation program to generate a set of manifest files in the assets directory.

The command to generate the manifest files displays a warning message before it consumes the **install-config.yaml** file.

If you plan to reuse the **install-config.yaml** file, create a backup copy of it before you back it up before you generate the manifest files.

Procedure

1. Optional: Create a backup copy of the **install-config.yaml** file:

```
$ cp install-config.yaml install-config.yaml.backup
```

2. Generate a set of manifests in your assets directory:

```
$ openshift-install create manifests --dir $ASSETS_DIR
```

This command displays the following messages.

Example output

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

The command generates the following manifest files:

Example output

```
$ tree
.
├── wrk
│   ├── manifests
│   │   ├── 04-openshift-machine-config-operator.yaml
│   │   ├── cluster-config.yaml
│   │   ├── cluster-dns-02-config.yml
│   │   ├── cluster-infrastructure-02-config.yml
│   │   ├── cluster-ingress-02-config.yml
│   │   ├── cluster-network-01-crd.yml
│   │   ├── cluster-network-02-config.yml
│   │   ├── cluster-proxy-01-config.yaml
│   │   ├── cluster-scheduler-02-config.yml
│   │   ├── cvo-overrides.yaml
│   │   ├── etcd-ca-bundle-configmap.yaml
│   │   ├── etcd-client-secret.yaml
│   │   ├── etcd-host-service-endpoints.yaml
│   │   ├── etcd-host-service.yaml
│   │   ├── etcd-metric-client-secret.yaml
│   │   ├── etcd-metric-serving-ca-configmap.yaml
│   │   ├── etcd-metric-signer-secret.yaml
│   │   ├── etcd-namespace.yaml
│   │   ├── etcd-service.yaml
│   │   ├── etcd-serving-ca-configmap.yaml
│   │   ├── etcd-signer-secret.yaml
│   │   ├── kube-cloud-config.yaml
│   │   ├── kube-system-configmap-root-ca.yaml
│   │   ├── machine-config-server-tls-secret.yaml
│   │   └── openshift-config-secret-pull-secret.yaml
│   └── openshift
│       ├── 99_kubeadmin-password-secret.yaml
│       ├── 99_openshift-cluster-api_master-user-data-secret.yaml
│       ├── 99_openshift-cluster-api_worker-user-data-secret.yaml
│       ├── 99_openshift-machineconfig_99-master-ssh.yaml
│       ├── 99_openshift-machineconfig_99-worker-ssh.yaml
│       └── openshift-install-manifests.yaml
```

Next steps

- Make control plane nodes non-schedulable.

22.5.18. Making control-plane nodes non-schedulable

Because you are manually creating and deploying the control plane machines, you must configure a manifest file to make the control plane nodes non-schedulable.

Procedure

1. To make the control plane nodes non-schedulable, enter:

```
$ python3 -c 'import os, yaml
path = "%s/manifests/cluster-scheduler-02-config.yml" % os.environ["ASSETS_DIR"]
data = yaml.safe_load(open(path))
data["spec"]["mastersSchedulable"] = False
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

22.5.19. Building the Ignition files

To build the Ignition files from the manifest files you just generated and modified, you run the installation program. This action creates a Red Hat Enterprise Linux CoreOS (RHCOS) machine, **initramfs**, which fetches the Ignition files and performs the configurations needed to create a node.

In addition to the Ignition files, the installation program generates the following:

- An **auth** directory that contains the admin credentials for connecting to the cluster with the **oc** and **kubectrl** utilities.
- A **metadata.json** file that contains information such as the OpenShift Container Platform cluster name, cluster ID, and infrastructure ID for the current installation.

The Ansible playbooks for this installation process use the value of **infralD** as a prefix for the virtual machines they create. This prevents naming conflicts when there are multiple installations in the same oVirt/RHV cluster.



NOTE

Certificates in Ignition configuration files expire after 24 hours. Complete the cluster installation and keep the cluster running in a non-degraded state for 24 hours so that the first certificate rotation can finish.

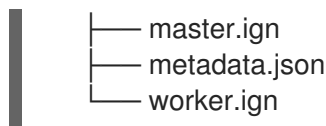
Procedure

1. To build the Ignition files, enter:

```
$ openshift-install create ignition-configs --dir $ASSETS_DIR
```

Example output

```
$ tree
.
├── wrk
│   ├── auth
│   │   ├── kubeadmin-password
│   │   └── kubeconfig
│   └── bootstrap.ign
```



22.5.20. Creating templates and virtual machines

After confirming the variables in the **inventory.yml**, you run the first Ansible provisioning playbook, **create-templates-and-vms.yml**.

This playbook uses the connection parameters for the RHV Manager from **\$HOME/.ovirt/ovirt-config.yml** and reads **metadata.json** in the assets directory.

If a local Red Hat Enterprise Linux CoreOS (RHCOS) image is not already present, the playbook downloads one from the URL you specified for **image_url** in **inventory.yml**. It extracts the image and uploads it to RHV to create templates.

The playbook creates a template based on the **control_plane** and **compute** profiles in the **inventory.yml** file. If these profiles have different names, it creates two templates.

When the playbook finishes, the virtual machines it creates are stopped. You can get information from them to help configure other infrastructure elements. For example, you can get the virtual machines' MAC addresses to configure DHCP to assign permanent IP addresses to the virtual machines.

Procedure

1. In **inventory.yml**, under the **control_plane** and **compute** variables, change both instances of **type: high_performance** to **type: server**.
2. Optional: If you plan to perform multiple installations to the same cluster, create different templates for each OpenShift Container Platform installation. In the **inventory.yml** file, prepend the value of **template** with **infraID**. For example:

```

control_plane:
  cluster: "{{ ovirt_cluster }}"
  memory: 16GiB
  sockets: 4
  cores: 1
  template: "{{ metadata.infraID }}-rhcos_tpl"
  operating_system: "rhcos_x64"
  ...

```

3. Create the templates and virtual machines:

```
$ ansible-playbook -i inventory.yml create-templates-and-vms.yml
```

22.5.21. Creating the bootstrap machine

You create a bootstrap machine by running the **bootstrap.yml** playbook. This playbook starts the bootstrap virtual machine, and passes it the **bootstrap.ign** Ignition file from the assets directory. The bootstrap node configures itself so it can serve Ignition files to the control plane nodes.

To monitor the bootstrap process, you use the console in the RHV Administration Portal or connect to the virtual machine by using SSH.

Procedure

Procedure

1. Create the bootstrap machine:

```
$ ansible-playbook -i inventory.yml bootstrap.yml
```

2. Connect to the bootstrap machine using a console in the Administration Portal or SSH. Replace **<bootstrap_ip>** with the bootstrap node IP address. To use SSH, enter:

```
$ ssh core@<bootstrap.ip>
```

3. Collect **bootkube.service** journald unit logs for the release image service from the bootstrap node:

```
[core@ocp4-1k6b4-bootstrap ~]$ journalctl -b -f -u release-image.service -u bootkube.service
```

**NOTE**

The **bootkube.service** log on the bootstrap node outputs etcd **connection refused** errors, indicating that the bootstrap server is unable to connect to etcd on control plane nodes. After etcd has started on each control plane node and the nodes have joined the cluster, the errors should stop.

22.5.22. Creating the control plane nodes

You create the control plane nodes by running the **masters.yml** playbook. This playbook passes the **master.ign** Ignition file to each of the virtual machines. The Ignition file contains a directive for the control plane node to get the Ignition from a URL such as <https://api-int.ocp4.example.org:22623/config/master>. The port number in this URL is managed by the load balancer, and is accessible only inside the cluster.

Procedure

1. Create the control plane nodes:

```
$ ansible-playbook -i inventory.yml masters.yml
```

2. While the playbook creates your control plane, monitor the bootstrapping process:

```
$ openshift-install wait-for bootstrap-complete --dir $ASSETS_DIR
```

Example output

```
INFO API v1.25.0 up
INFO Waiting up to 40m0s for bootstrapping to complete...
```

3. When all the pods on the control plane nodes and etcd are up and running, the installation program displays the following output.

Example output

```
INFO It is now safe to remove the bootstrap resources
```


22.5.23. Verifying cluster status

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=$ASSETS_DIR/auth/kubeconfig
```

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

3. View your cluster's version:

```
$ oc get clusterversion
```

4. View your Operators' status:

```
$ oc get clusteroperator
```

5. View all running pods in the cluster:

```
$ oc get pods -A
```

22.5.24. Removing the bootstrap machine

After the **wait-for** command shows that the bootstrap process is complete, you must remove the bootstrap virtual machine to free up compute, memory, and storage resources. Also, remove settings for the bootstrap machine from the load balancer directives.

Procedure

1. To remove the bootstrap machine from the cluster, enter:

```
$ ansible-playbook -i inventory.yml retire-bootstrap.yml
```

2. Remove settings for the bootstrap machine from the load balancer directives.

22.5.25. Creating the worker nodes and completing the installation

Creating worker nodes is similar to creating control plane nodes. However, worker nodes workers do not automatically join the cluster. To add them to the cluster, you review and approve the workers' pending CSRs (Certificate Signing Requests).

After approving the first requests, you continue approving CSR until all of the worker nodes are approved. When you complete this process, the worker nodes become **Ready** and can have pods scheduled to run on them.

Finally, monitor the command line to see when the installation process completes.

Procedure

1. Create the worker nodes:

```
$ ansible-playbook -i inventory.yml workers.yml
```

2. To list all of the CSRs, enter:

```
$ oc get csr -A
```

Eventually, this command displays one CSR per node. For example:

Example output

```
NAME      AGE  SIGNERNAME                                REQUESTOR
CONDITION
csr-2lnxd 63m  kubernetes.io/kubelet-serving             system:node:ocp4-1k6b4-
master0.ocp4.example.org                 Approved,Issued
csr-hff4q 64m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-hsn96 60m  kubernetes.io/kubelet-serving             system:node:ocp4-1k6b4-
master2.ocp4.example.org                 Approved,Issued
csr-m724n 6m2s kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
csr-p4dz2 60m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-t9vfj 60m  kubernetes.io/kubelet-serving             system:node:ocp4-1k6b4-
master1.ocp4.example.org                 Approved,Issued
csr-tggtr 61m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Approved,Issued
csr-wcbrf 7m6s kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
```

3. To filter the list and see only pending CSRs, enter:

```
$ watch "oc get csr -A | grep pending -i"
```

This command refreshes the output every two seconds and displays only pending CSRs. For example:

Example output

```
Every 2.0s: oc get csr -A | grep pending -i

csr-m724n 10m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
csr-wcbrf 11m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
```

- Inspect each pending request. For example:

Example output

```
$ oc describe csr csr-m724n
```

Example output

```
Name:          csr-m724n
Labels:        <none>
Annotations:   <none>
CreationTimestamp: Sun, 19 Jul 2020 15:59:37 +0200
Requesting User: system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper
Signer:        kubernetes.io/kube-apiserver-client-kubelet
Status:        Pending
Subject:
  Common Name:  system:node:ocp4-1k6b4-worker1.ocp4.example.org
  Serial Number:
  Organization: system:nodes
Events: <none>
```

- If the CSR information is correct, approve the request:

```
$ oc adm certificate approve csr-m724n
```

- Wait for the installation process to finish:

```
$ openshift-install wait-for install-complete --dir $ASSETS_DIR --log-level debug
```

When the installation completes, the command line displays the URL of the OpenShift Container Platform web console and the administrator user name and password.

22.5.26. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

22.5.27. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

22.6. UNINSTALLING A CLUSTER ON RHV

You can remove an OpenShift Container Platform cluster from Red Hat Virtualization (RHV).

22.6.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$ ./openshift-install destroy cluster \
  --dir <installation_directory> --log-level info 1 2
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2** To view different details, specify **warn**, **debug**, or **error** instead of **info**.

**NOTE**

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

22.6.2. Removing a cluster that uses user-provisioned infrastructure

When you are finished using the cluster, you can remove a cluster that uses user-provisioned infrastructure from your cloud.

Prerequisites

- Have the original playbook files, assets directory and files, and **\$ASSETS_DIR** environment variable that you used to you install the cluster. Typically, you can achieve this by using the same computer you used when you installed the cluster.

Procedure

1. To remove the cluster, enter:

```
$ ansible-playbook -i inventory.yml \  
  retire-bootstrap.yml \  
  retire-masters.yml \  
  retire-workers.yml
```

2. Remove any configurations you added to DNS, load balancers, and any other infrastructure for this cluster.

CHAPTER 23. INSTALLING ON VSPHERE

23.1. PREPARING TO INSTALL ON VSPHERE

23.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall and plan to use Telemetry, you [configured the firewall to allow the sites](#) required by your cluster.
- You reviewed your VMware platform licenses. Red Hat does not place any restrictions on your VMware licenses, but some VMware infrastructure components require licensing.

23.1.2. Choosing a method to install OpenShift Container Platform on vSphere

You can install OpenShift Container Platform with the [Assisted Installer](#). This method requires no setup for the installer, and is ideal for connected environments like vSphere. Installing with the Assisted Installer also provides integration with vSphere, enabling autoscaling. See [Installing an on-premise cluster using the Assisted Installer](#) for additional details.

You can also install OpenShift Container Platform on vSphere by using installer-provisioned or user-provisioned infrastructure. Installer-provisioned infrastructure is ideal for installing in environments with air-gapped/restricted networks, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provide. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See the [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

23.1.2.1. Installer-provisioned infrastructure installation of OpenShift Container Platform on vSphere

Installer-provisioned infrastructure allows the installation program to preconfigure and automate the provisioning of resources required by OpenShift Container Platform.

- [Installing a cluster on vSphere](#) You can install OpenShift Container Platform on vSphere by using installer-provisioned infrastructure installation with no customization.

- **Installing a cluster on vSphere with customizations** You can install OpenShift Container Platform on vSphere by using installer-provisioned infrastructure installation with the default customization options.
- **Installing a cluster on vSphere with network customizations** You can install OpenShift Container Platform on installer-provisioned vSphere infrastructure, with network customizations. You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- **Installing a cluster on vSphere in a restricted network** You can install a cluster on VMware vSphere infrastructure in a restricted network by creating an internal mirror of the installation release content. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.

23.1.2.2. User-provisioned infrastructure installation of OpenShift Container Platform on vSphere

User-provisioned infrastructure requires the user to provision all resources required by OpenShift Container Platform.

- **Installing a cluster on vSphere with user-provisioned infrastructure** You can install OpenShift Container Platform on VMware vSphere infrastructure that you provision.
- **Installing a cluster on vSphere with network customizations with user-provisioned infrastructure:** You can install OpenShift Container Platform on VMware vSphere infrastructure that you provision with customized network configuration options.
- **Installing a cluster on vSphere in a restricted network with user-provisioned infrastructure** OpenShift Container Platform can be installed on VMware vSphere infrastructure that you provision in a restricted network.

23.1.3. VMware vSphere infrastructure requirements

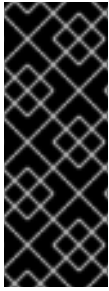
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.1. Version requirements for vSphere virtual environments

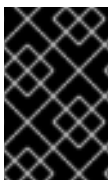
Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

**IMPORTANT**

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.2. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.1.4. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later

- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).

23.1.5. Configuring the vSphere connection settings

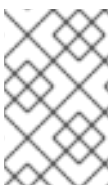
- [Updating the vSphere connection settings following an installation](#) For installations on vSphere using the Assisted Installer, you must manually update the vSphere connection settings to complete the installation. For installer-provisioned or user-provisioned infrastructure installations on vSphere, you can optionally validate or modify the vSphere connection settings at any time.

23.1.6. Uninstalling an installer-provisioned infrastructure installation of OpenShift Container Platform on vSphere

- [Uninstalling a cluster on vSphere that uses installer-provisioned infrastructure](#) You can remove a cluster that you deployed on VMware vSphere infrastructure that used installer-provisioned infrastructure.

23.2. INSTALLING A CLUSTER ON VSPHERE

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure.



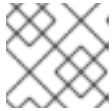
NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

23.2.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).

- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

23.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.2.3. VMware vSphere infrastructure requirements

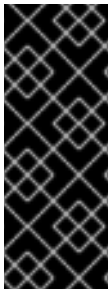
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.3. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

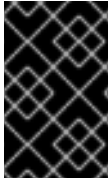


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.4. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.2.4. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 23.5. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.6. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.7. Ports used for control plane machine to control plane machine communications

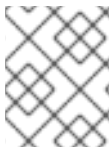
Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

23.2.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.2.6. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder

to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 23.1. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	

vSphere object for role	When required	InventoryService.Tagging.O bject.Attachable Resource.AssignVMToPool API VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone VirtualMachine.Provisionin g.MarkAsTemplate VirtualMachine.Provisionin g.DeployTemplate

vSphere object for role	When required	Required privileges in vSphere
	<p>The installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.</p>	<p>APIAttachableResource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning</p>

vSphere object for role	When required	Required privileges in vSphere API

Example 23.2. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration". "Add new disk"
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration". "Add existing disk" "Virtual machine"."Change Configuration". "Add new disk" "Virtual machine"."Change Configuration". "Add or remove device" "Virtual machine"."Change Configuration". "Advanced configuration" "Virtual machine"."Change Configuration". "Set annotation" "Virtual machine"."Change Configuration". "Change CPU count" "Virtual machine"."Change Configuration". "Extend virtual disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Acquire Required privileges in vCenter GUI "Virtual machine". "Change "Virtual machine". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit Inventory". "Create from existing" "Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Mark as template" "Virtual machine". Provisioning. "Deploy template"
vSphere vCenter Datacenter	If the installation program	"vSphere Tagging". "Assign

vSphere object for role	creates the virtual machine When required Folder, FC/UP, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete	or Unassign vSphere Tag Required privileges in vCenter GUI Resource."Assign virtual
	privileges are optional if your cluster does not use the Machine API.	machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change Configuration".Rename "Virtual machine"."Change Configuration"."Reset guest information" "Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system

vSphere object for role	When required	management by VIX API" Required privileges in vCenter GUI "Virtual machine".Interaction."Power off" "Virtual machine".Interaction."Power on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clone virtual machine" "Virtual machine".Provisioning."Deploy template" "Virtual machine".Provisioning."Mark as template" Folder."Create folder" Folder."Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.3. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category

- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.8. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

23.2.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **`./openshift-install gather`** command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:


```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

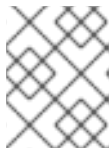
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

23.2.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.



IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

23.2.9. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

- From the vCenter home page, download the vCenter’s root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
- Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl
```

3 directories, 15 files

- Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

- Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

23.2.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Provide values at the prompts:
 - a. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- b. Select **vsphere** as the platform to target.

- c. Specify the name of your vCenter instance.
- d. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.



IMPORTANT

Some VMware vCenter Single Sign-On (SSO) environments with Active Directory (AD) integration might primarily require you to use the traditional login method, which requires the **<domain>** construct.

To ensure that vCenter account permission checks complete properly, consider using the User Principal Name (UPN) login method, such as **<username>@<fully_qualified_domainname>**.

- e. Select the data center in your vCenter instance to connect to.
- f. Select the default vCenter datastore to use.



NOTE

Datastore and cluster names cannot exceed 60 characters; therefore, ensure the combined string length does not exceed the 60 character limit.

- g. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
- h. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
- i. Enter the virtual IP address that you configured for control plane API access.
- j. Enter the virtual IP address that you configured for cluster ingress.
- k. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
- l. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.



NOTE

Datastore and cluster names cannot exceed 60 characters; therefore, ensure the combined string length does not exceed the 60 character limit.

- m. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

23.2.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

23.2.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```


Example output

```
system:admin
```

23.2.13. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

23.2.13.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

23.2.13.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

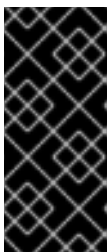
Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.2.13.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

-

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

23.2.13.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.

4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.2.14. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.2.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use](#)

[subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.2.16. Next steps

- [Customize your cluster.](#)
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.3. INSTALLING A CLUSTER ON VSPHERE WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

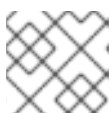


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

23.3.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

23.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.3.3. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.9. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

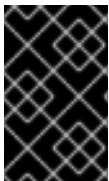


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.10. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.3.4. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 23.11. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests

Protocol	Port	Description
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.12. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.13. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

23.3.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later

- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.3.6. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 23.4. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API VirtualMachine.Config.AdvancedConfig
		VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk

vSphere object for role	delete privileges are optional if your cluster does not use the Machine API. When required	Required privileges in vSphere API
		VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 23.5. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
		<p>"Virtual machine"."Change Configuration".Add or remove device"</p> <p>"Virtual machine"."Change Configuration"."Advanced configuration"</p> <p>"Virtual machine"."Change Configuration"."Set annotation"</p> <p>"Virtual machine"."Change Configuration"."Change CPU count"</p> <p>"Virtual machine"."Change Configuration"."Extend virtual disk"</p> <p>"Virtual machine"."Change Configuration"."Acquire disk lease"</p> <p>"Virtual machine"."Change Configuration"."Modify device settings"</p> <p>"Virtual machine"."Change Configuration"."Change Memory"</p> <p>"Virtual machine"."Change Configuration"."Remove disk"</p> <p>"Virtual machine"."Change Configuration".Rename</p> <p>"Virtual machine"."Change Configuration"."Reset guest information"</p> <p>"Virtual machine"."Change Configuration"."Change resource"</p> <p>"Virtual machine"."Change Configuration"."Change Settings"</p> <p>"Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility"</p> <p>"Virtual machine".Interaction."Guest operating system management by VIX API"</p> <p>"Virtual machine".Interaction."Power off"</p> <p>"Virtual machine".Interaction."Power on"</p> <p>"Virtual machine".Interaction.Reset</p> <p>"Virtual machine"."Edit Inventory"."Create new"</p>

vSphere object for role	When required	"Virtual machine". "Edit Inventory". "Create from existing"
		"Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Deploy template" "Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.6. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 23.14. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

23.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

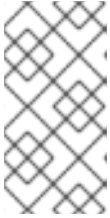
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

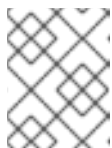
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

23.3.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.



IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

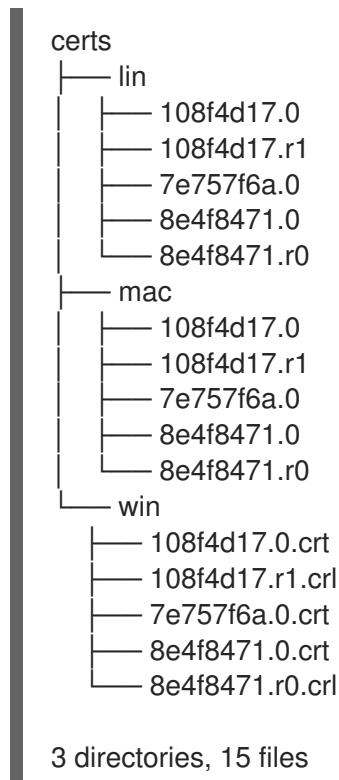
5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

23.3.9. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

23.3.10. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.15. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

23.3.11. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

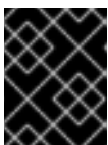
- b. At the prompts, provide the configuration details for your cloud:

- i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
 - iii. Specify the name of your vCenter instance.
 - iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
 - v. Select the data center in your vCenter instance to connect to.
 - vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

23.3.11.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

23.3.11.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 23.16. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>.<baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code> . For additional information about platform . <code><platform></code> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

23.3.11.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 23.17. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>

Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

23.3.11.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 23.18. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String

Parameter	Description	Values
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string (<code>""</code>).

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 592 1393" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1442 592 1637" style="background-color: #f0f0f0; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

23.3.11.1.4. Additional VMware vSphere configuration parameters


Additional VMware vSphere configuration parameters are described in the following table.




NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.19. Additional VMware vSphere cluster parameters

Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String
apiVIPs	The virtual IP (VIP) address that you configured for control plane API access.  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .

Parameter	Description	Values
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

23.3.11.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

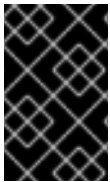
Table 23.20. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer

Parameter	Description	Values
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

23.3.11.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.21. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String

Parameter	Description	Values
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String

23.3.11.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

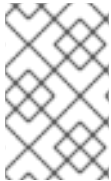
```

apiVersion: v1
baseDomain: example.com ❶
compute: ❷
  name: worker
  replicas: 3
  platform:
    vsphere: ❸
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: ❹
  name: master
  replicas: 3
  platform:
    vsphere: ❺
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster ❻
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    resourcePool: resource_pool ❼
    diskType: thin ❽
    network: VM_Network
    cluster: vsphere_cluster_name ❾
    apiVIPs:
      - api_vip
    ingressVIPs:
      - ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'

```

- ❶ The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- ❷❹ The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

- 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
- 6 The cluster name that you specified in your DNS records.
- 7 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
- 8 The vSphere disk provisioning method.
- 9 The vSphere cluster to install the OpenShift Container Platform cluster in.



NOTE

In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

23.3.11.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```



```

additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.3.11.4. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).



IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

■

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

- To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

- To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

- Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample `install-config.yaml` file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
```

```

datacenter: datacenter 7
defaultDatastore: datastore 8
folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
cluster: cluster 10
resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
diskType: thin
failureDomains: 12
- name: <machine_pool_zone_1> 13
  region: <region_tag_1> 14
  zone: <zone_tag_1> 15
  topology: 16
    datacenter: <datacenter1> 17
    computeCluster: "/<datacenter1>/host/<cluster1>" 18
    resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
    networks: 20
    - <VM_Network1_name>
    datastore: "/<datacenter1>/datastore/<datastore1>" 21
- name: <machine_pool_zone_2>
  region: <region_tag_2>
  zone: <zone_tag_2>
  topology:
    datacenter: <datacenter2>
    computeCluster: "/<datacenter2>/host/<cluster2>"
    networks:
    - <VM_Network2_name>
    datastore: "/<datacenter2>/datastore/<datastore2>"
    resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
    folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1** You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2** **3** An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4** **5** **6** **7** **8** **9** **10** **11** The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13** Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14** You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15** You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.

- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.3.12. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

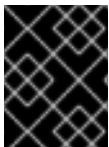
**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

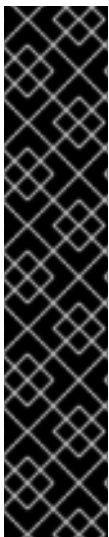
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

23.3.13. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

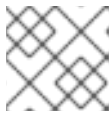
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

23.3.14. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```


- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

23.3.15. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

23.3.15.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from **Removed** to **Managed**. When this has completed, you must configure storage.

23.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

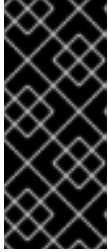
Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.3.15.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

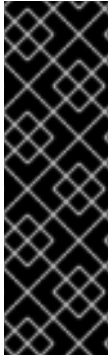
Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.

**NOTE**

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```

**NOTE**

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

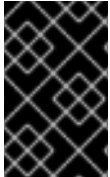
```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

23.3.15.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
```

```
- ReadWriteOnce 3
resources:
  requests:
    storage: 100Gi 4
```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.3.16. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.

4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.3.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.3.18. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

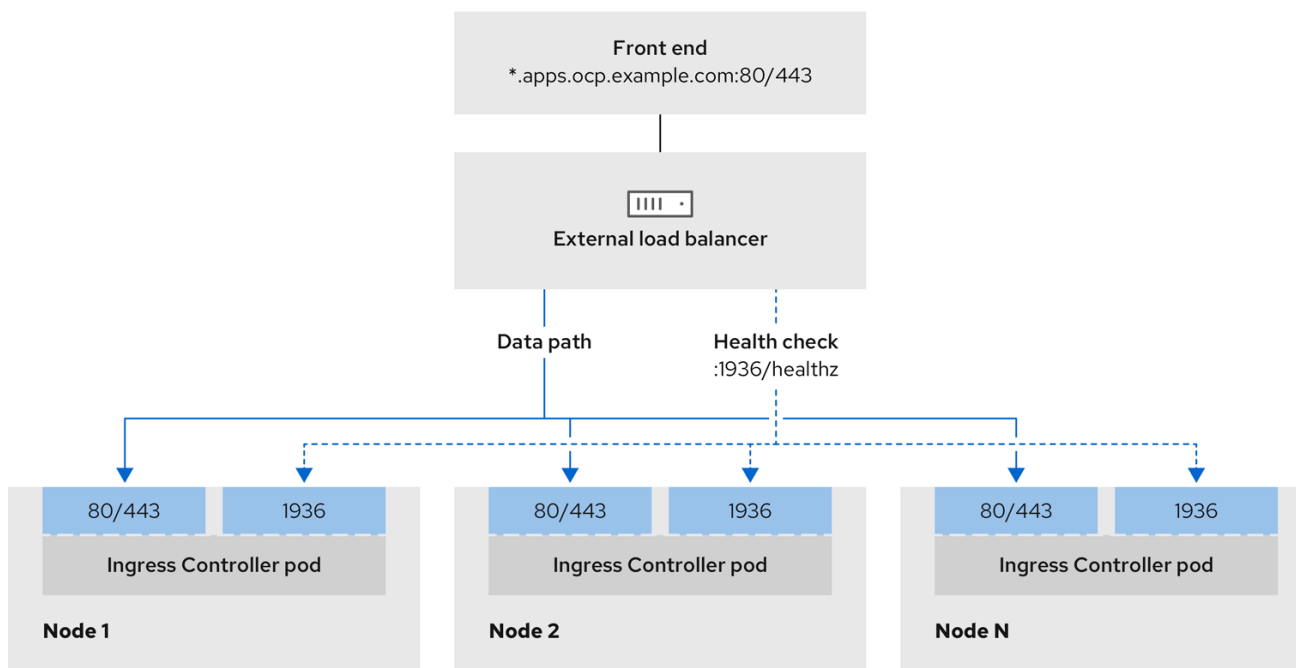
The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

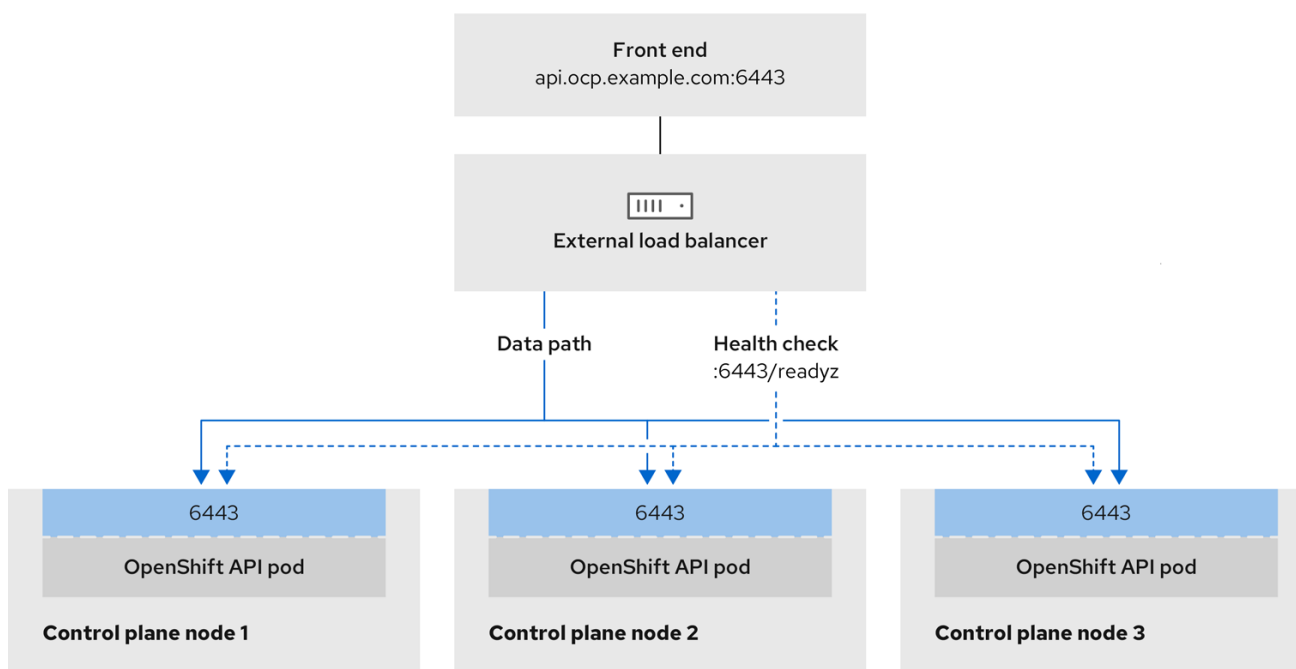
You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 23.1. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



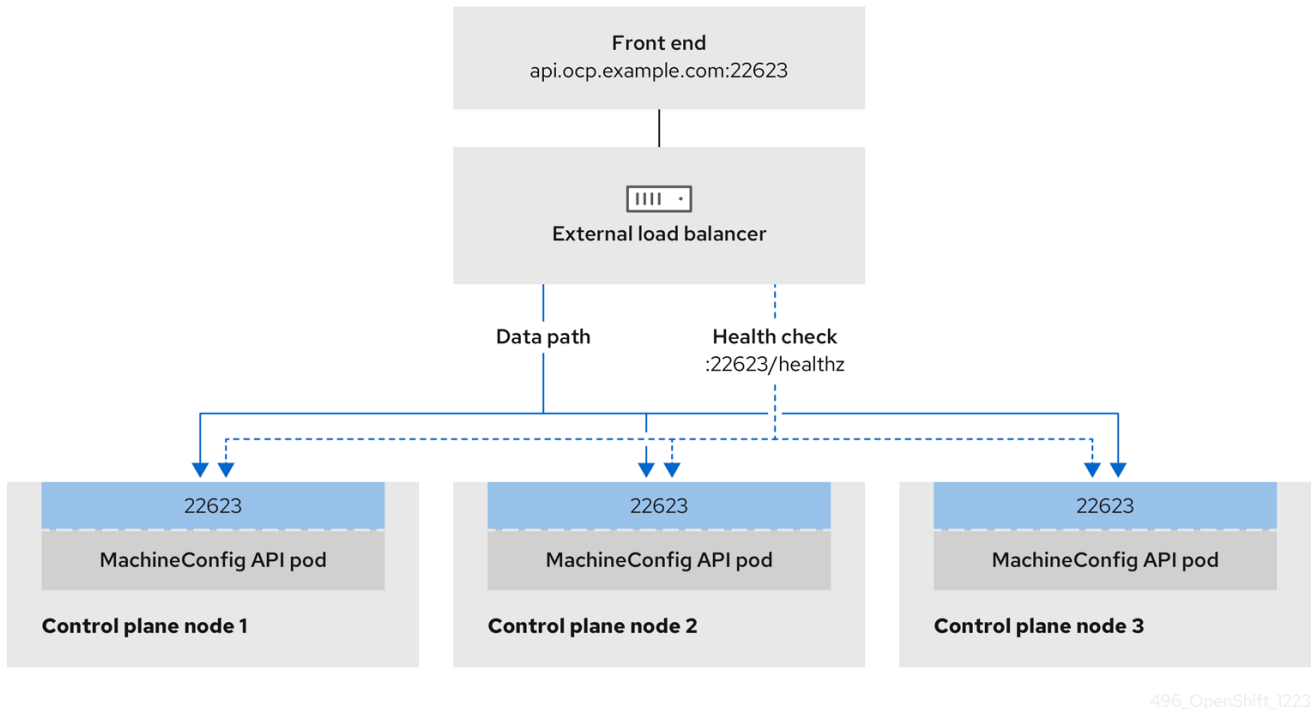
496_OpenShift_I223

Figure 23.2. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment



496_OpenShift_I223

Figure 23.3. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

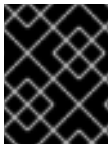
Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

23.3.18.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
}
```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKIh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private

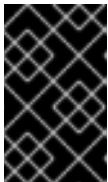
```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

23.3.19. Next steps

- [Customize your cluster.](#)

- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.4. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

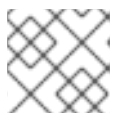


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

23.4.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, confirm with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

23.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.4.3. VMware vSphere infrastructure requirements

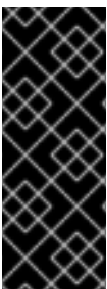
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.22. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.23. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.4.4. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 23.24. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests

Protocol	Port	Description
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.25. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.26. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

23.4.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later

- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.4.6. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 23.7. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API
		VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk

vSphere object for role	delete privileges are optional if your cluster does not use the Machine API. When required	VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 23.8. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration".Change resource" "Virtual machine"."Change Configuration".Change Settings" "Virtual machine"."Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory".Create new" "Virtual machine"."Edit Inventory".Create from existing" "Virtual machine"."Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource.Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration".Add existing disk" "Virtual machine"."Change Configuration".Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
		<p>"Virtual machine"."Change Configuration". "Add or remove device"</p> <p>"Virtual machine"."Change Configuration"."Advanced configuration"</p> <p>"Virtual machine"."Change Configuration"."Set annotation"</p> <p>"Virtual machine"."Change Configuration"."Change CPU count"</p> <p>"Virtual machine"."Change Configuration"."Extend virtual disk"</p> <p>"Virtual machine"."Change Configuration"."Acquire disk lease"</p> <p>"Virtual machine"."Change Configuration"."Modify device settings"</p> <p>"Virtual machine"."Change Configuration"."Change Memory"</p> <p>"Virtual machine"."Change Configuration"."Remove disk"</p> <p>"Virtual machine"."Change Configuration".Rename</p> <p>"Virtual machine"."Change Configuration"."Reset guest information"</p> <p>"Virtual machine"."Change Configuration"."Change resource"</p> <p>"Virtual machine"."Change Configuration"."Change Settings"</p> <p>"Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility"</p> <p>"Virtual machine".Interaction."Guest operating system management by VIX API"</p> <p>"Virtual machine".Interaction."Power off"</p> <p>"Virtual machine".Interaction."Power on"</p> <p>"Virtual machine".Interaction.Reset</p> <p>"Virtual machine"."Edit Inventory"."Create new"</p>

vSphere object for role	When required	"Virtual machine". "Edit Inventory". "Create from Existing" "Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Deploy template" "Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.9. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 23.27. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

23.4.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

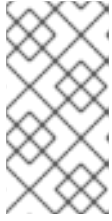
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

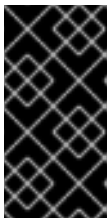
- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

23.4.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

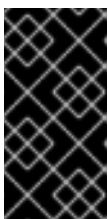


IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

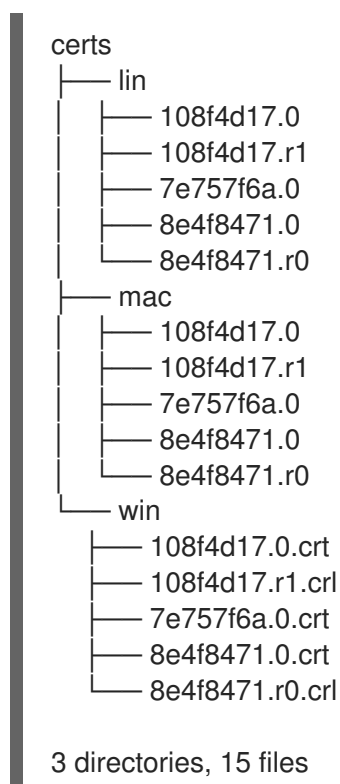
5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

23.4.9. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

23.4.10. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.28. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

23.4.11. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

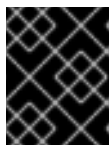
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
 - iii. Specify the name of your vCenter instance.
 - iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
 - v. Select the data center in your vCenter instance to connect to.
 - vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

23.4.11.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

23.4.11.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 23.29. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or <code>{}</code> . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

23.4.11.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 23.30. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>

Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>



23.4.11.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 23.31. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String

Parameter	Description	Values
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 150px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 20px;"> <div style="width: 60px; height: 80px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.

Parameter	Description	Values
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 100px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

23.4.11.1.4. Additional VMware vSphere configuration parameters


Additional VMware vSphere configuration parameters are described in the following table.




NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.32. Additional VMware vSphere cluster parameters

Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String
apiVIPs	The virtual IP (VIP) address that you configured for control plane API access.  NOTE In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.	An IP address, for example 128.0.0.1 .

Parameter	Description	Values
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

23.4.11.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.33. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer

Parameter	Description	Values
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

23.4.11.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.34. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String

Parameter	Description	Values
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String

23.4.11.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 3
  platform:
    vsphere: 3
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: 4
  name: master
  replicas: 3
  platform:
    vsphere: 5
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster 6
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 7
  serviceNetwork:
    - 172.30.0.0/16
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    resourcePool: resource_pool 8
    diskType: thin 9
    network: VM_Network
    cluster: vsphere_cluster_name 10
  apiVIPs:
    - api_vip
  ingressVIPs:
    - ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```


-
- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
 - 2 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
 - 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
 - 6 The cluster name that you specified in your DNS records.
 - 8 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
 - 9 The vSphere disk provisioning method.
 - 10 The vSphere cluster to install the OpenShift Container Platform cluster in.
 - 7 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.



NOTE

In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

23.4.11.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.4.11.4. Configuring regions and zones for a VMware vCenter

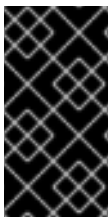
You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

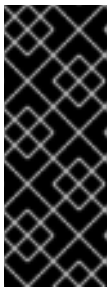


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
vsphere:
  zones: 2
  - "<machine_pool_zone_1>"
  - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
vsphere:
  zones: 3
```

```

- "<machine_pool_zone_1>"
- "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> 4
    username: <username> 5
    password: <password> 6
    datacenter: datacenter 7
    defaultDatastore: datastore 8
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
    cluster: cluster 10
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
    diskType: thin
    failureDomains: 12
    - name: <machine_pool_zone_1> 13
      region: <region_tag_1> 14
      zone: <zone_tag_1> 15
      topology: 16
        datacenter: <datacenter1> 17
        computeCluster: "/<datacenter1>/host/<cluster1>" 18
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
        networks: 20
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>" 21
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.

- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.4.12. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

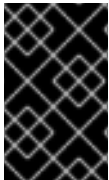
Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

23.4.13. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800

```

Enable IPsec for the OVN-Kubernetes network provider

```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}

```

- Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

23.4.14. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

23.4.14.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 23.35. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .


Field	Type	Description
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 23.36. **defaultNetwork** object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 23.37. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 23.38. ovnKubernetesConfig object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is 2⁽²³⁻¹⁴⁾=512.</p> <p>This field cannot be changed after installation.</p>	The default value is 100.64.0.0/16 .

Field	Type	Description
v6InternalSubnet	If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. This field cannot be changed after installation.	The default value is fd98::/48 .

Table 23.39. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 23.40. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 23.41. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

23.4.15. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

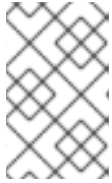
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```


-
- 1 For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



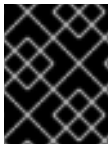
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to `<installation_directory>/openshift_install.log`.

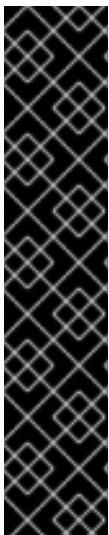


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

23.4.16. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.

To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

23.4.17. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

23.4.18. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

23.4.18.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

23.4.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

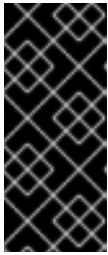
23.4.18.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.

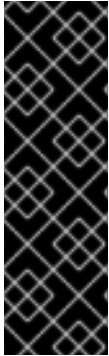
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

■

```
storage:
  pvc:
    claim: 1
```

- 1 Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

23.4.18.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
```

```
- ReadWriteOnce 3
resources:
  requests:
    storage: 100Gi 4
```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.4.19. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.

4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.4.20. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.4.21. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 23.4. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment

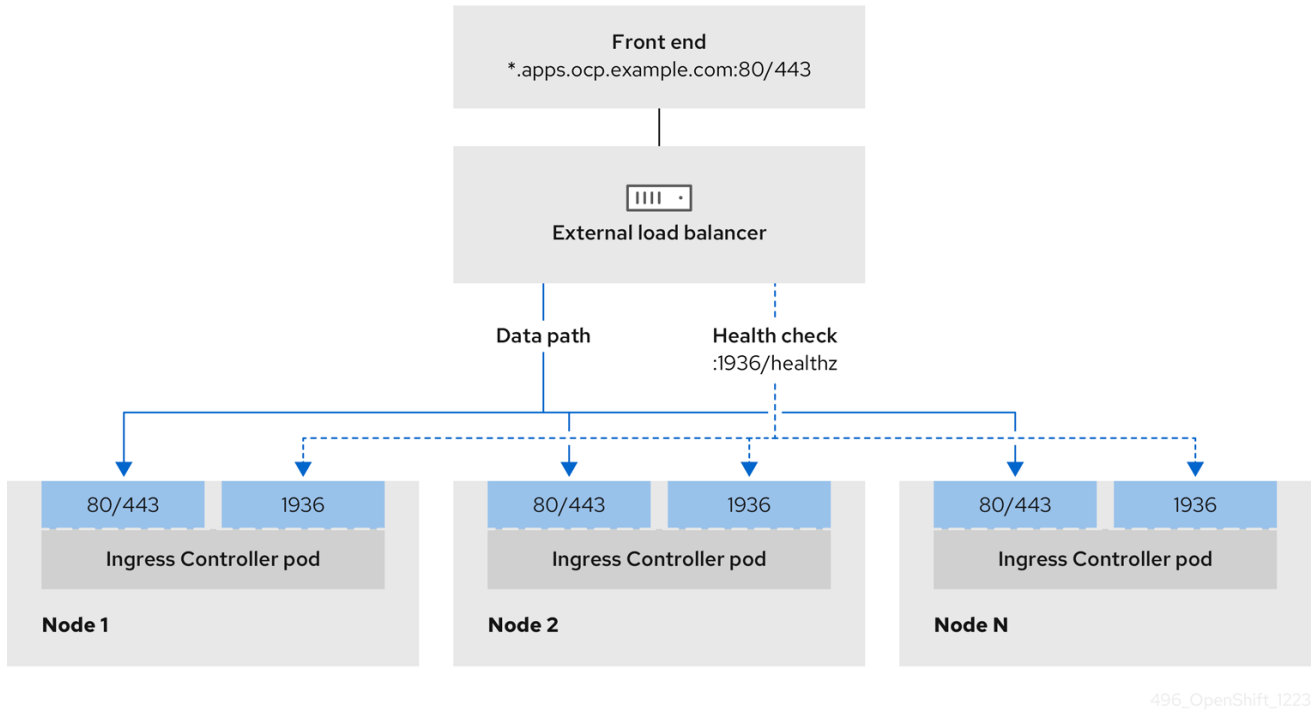


Figure 23.5. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment

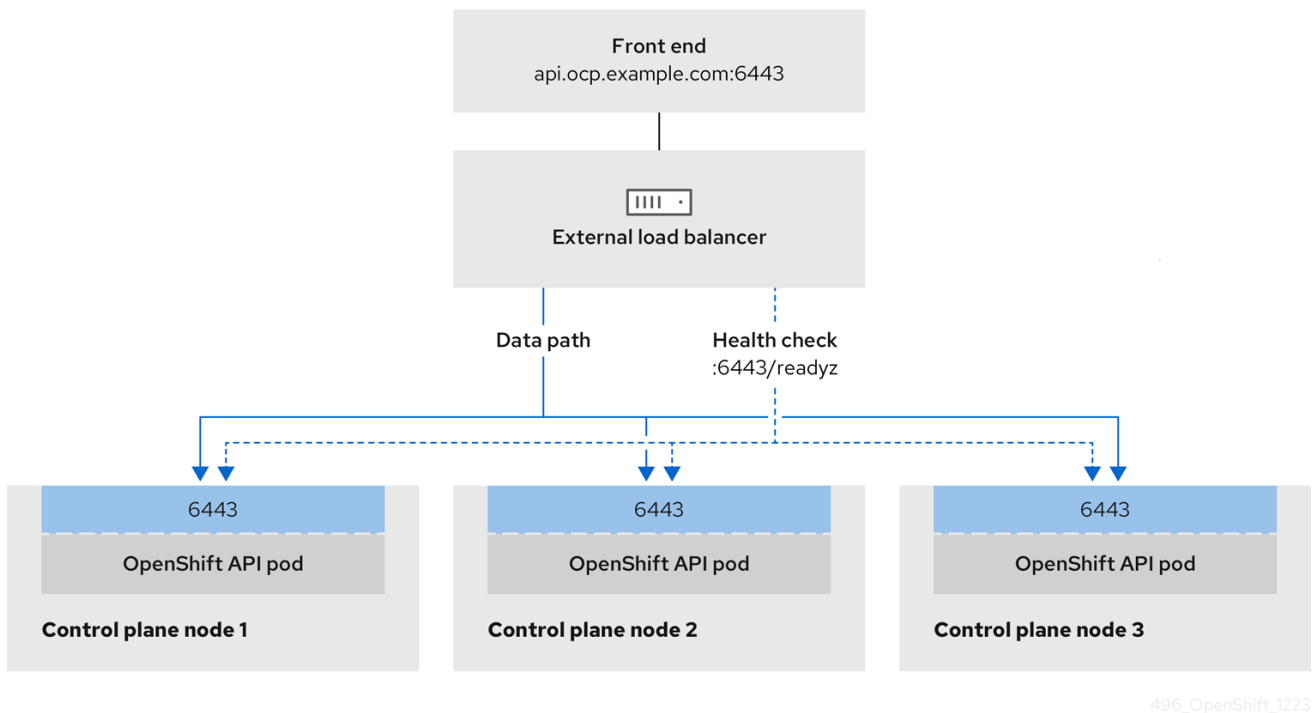
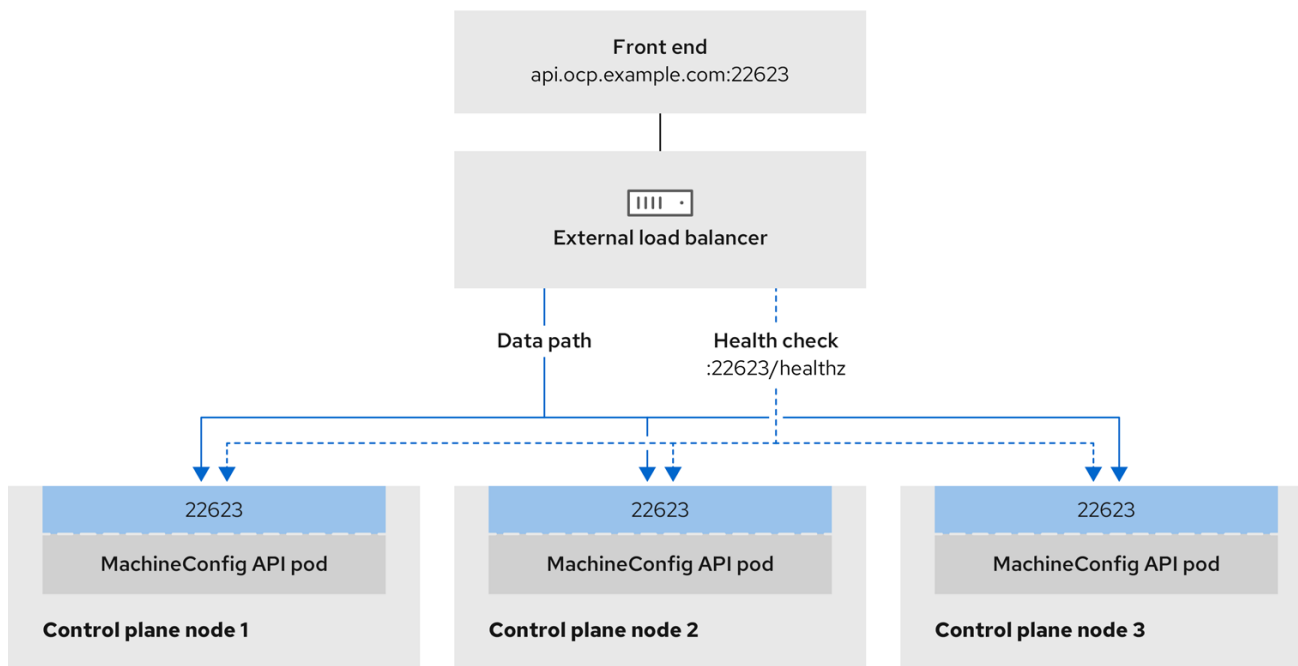


Figure 23.6. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



496_OpenShift_1223

The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

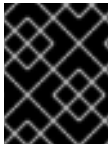
Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

23.4.21.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",

```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie: 1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/; HttpOnly; Secure; SameSite=None
cache-control: private

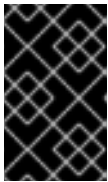
```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQWzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

23.4.22. Next steps

- [Customize your cluster.](#)

- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.5. INSTALLING A CLUSTER ON VSPHERE WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure that you provision.



NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

23.5.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.
- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

23.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.5.3. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.42. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

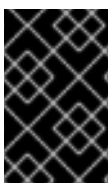


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.43. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.5.4. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.

**NOTE**

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.5.5. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

23.5.5.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 23.10. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk

vSphere object for role	When required	VirtualMachine.Config.Rename Required privileges in vSphere API
		VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.Rem

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 23.11. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Add or Remove device Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from existing virtual machine". "Edit virtual machine". "Remove" "Virtual machine". "Provisioning. "Clone virtual machine" "Virtual machine". "Provisioning. "Deploy template" "Virtual machine". "Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.12. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you

require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.44. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

- [Creating a compute machine set on vSphere](#)

23.5.5.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 23.45. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

23.5.5.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 23.46. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	Input/Output Per Second (IOPS)[1]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [2]	2	8 GB	100 GB	300

1. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so

you might need to over-allocate storage volume to obtain sufficient performance.

2. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

23.5.5.4. Certificate signing requests management

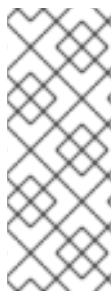
Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

23.5.5.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

23.5.5.5.1. Setting the cluster node hostnames through DHCP

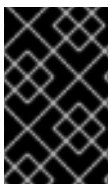
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

23.5.5.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 23.47. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.48. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.49. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00 to 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 to 00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

23.5.5.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.50. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <h4>IMPORTANT</h4> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

23.5.5.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 23.13. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4** Provides name resolution for the bootstrap machine.
- 5 6 7** Provides name resolution for the control plane machines.
- 8 9** Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 23.14. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1** Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2** Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3** Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6** Provides reverse DNS resolution for the control plane machines.

7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

23.5.5.7. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

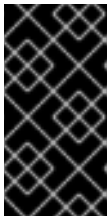


NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

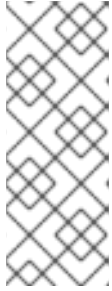
Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 23.51. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server

Port	Back-end machines (pool members)	Internal	External	Description
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

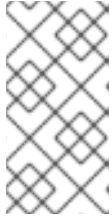
TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 23.52. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

23.5.5.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 23.15. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile /var/run/haproxy.pid
  maxconn 4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request 10s
  timeout queue 1m
  timeout connect 10s
  timeout client 1m
  timeout server 1m
  timeout http-keep-alive 10s
  timeout check 10s
  maxconn      3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s

```

```

fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

23.5.6. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

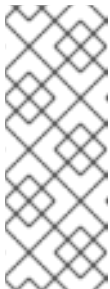
After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.

3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.



NOTE

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

23.5.7. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.



IMPORTANT

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

- From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

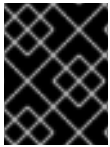
- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

23.5.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

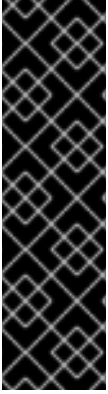
```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

23.5.9. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vccenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.53. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

23.5.10. Obtaining the installation program

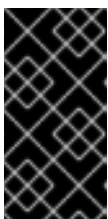
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

- Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
- Select your infrastructure provider.
- Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

- Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

- Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

23.5.11. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

- Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

- Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

23.5.11.1. Sample install-config.yaml file for VMware vSphere

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 13
    diskType: thin 14
  fips: false 15
  pullSecret: '{"auths": ...}' 16
  sshKey: 'ssh-ed25519 AAAA...' 17

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, (-), and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 3 You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.
- 5 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 6 The cluster name that you specified in your DNS records.
- 7 The fully-qualified hostname or IP address of the vCenter server.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

- 8 The name of the user for accessing the server.
- 9 The password associated with the vSphere user.
- 10 The vSphere datacenter.
- 11 The default vSphere datastore to use.
- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster and you do not want to use the default **StorageClass** object, named **thin**, you can omit the **folder** parameter from the **install-config.yaml** file.
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.
- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 The pull secret that you obtained from [OpenShift Cluster Manager Hybrid Cloud Console](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

23.5.11.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

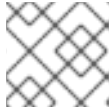
1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that can be used for HTTPS connections. The OpenShift Container Platform

that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

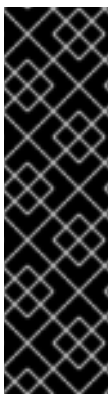


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.5.11.3. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).



IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
-
```



```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
    datacenter: datacenter ❼
    defaultDatastore: datastore ❽
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" ❾
    cluster: cluster ❿
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" ❶❶
    diskType: thin
    failureDomains: ❶❷
    - name: <machine_pool_zone_1> ❶❸
      region: <region_tag_1> ❶❹
      zone: <zone_tag_1> ❶❺
    topology: ❶❻
      datacenter: <datacenter1> ❶❼
      computeCluster: "/<datacenter1>/host/<cluster1>" ❶❶❶
      resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" ❶❶❷
    networks: ❶❷❶
    - <VM_Network1_name>
```

```

    datastore: "/<datacenter1>/datastore/<datastore1>" 21
- name: <machine_pool_zone_2>
  region: <region_tag_2>
  zone: <zone_tag_2>
  topology:
    datacenter: <datacenter2>
    computeCluster: "/<datacenter2>/host/<cluster2>"
    networks:
      - <VM_Network2_name>
    datastore: "/<datacenter2>/datastore/<datastore2>"
    resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
    folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**. If you do not specify a value, resources are installed in the root of the cluster **/example_datacenter/host/example_cluster/Resources**.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the

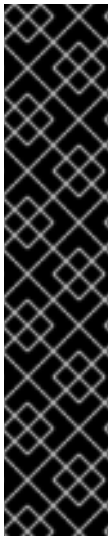
installation program uses the default vCenter topology.

- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.5.12. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
3. Check that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
 4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the `./<installation_directory>/auth` directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.5.13. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

23.5.14. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
}
```

```
"networkd": {},
"passwd": {},
"storage": {},
"systemd": {}
}
```

- 1 Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

- **<installation_directory>/master.ign**
- **<installation_directory>/worker.ign**
- **<installation_directory>/merge-bootstrap.ign**

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

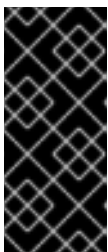
```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.



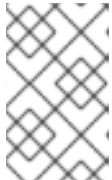
IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
 - a. Click the **VMs and Templates** view.

- b. Right-click the name of your datacenter.
 - c. Click **New Folder → New VM and Template Folder**.
 - d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.



NOTE

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

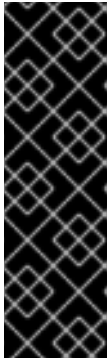
- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
 - Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
- g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.



IMPORTANT

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.

**IMPORTANT**

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone → Clone to Virtual Machine**
 - b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.

**NOTE**

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.

**IMPORTANT**

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```


- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

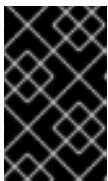
- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.
- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

23.5.15. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone → Clone to Virtual Machine**
2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**
7. On the **Customize hardware** tab, click **Advanced**.
 - a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

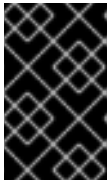
- Continue to create more compute machines for your cluster.

23.5.16. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

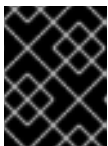
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions:** For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.



IMPORTANT

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

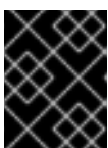
- **Retain existing partitions:** For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate **/var** partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers:** Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd:** Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var:** Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this

method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.

- 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

23.5.17. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

23.5.18. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1  
--log-level=info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

23.5.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

23.5.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

23.5.21. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

23.5.21.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

23.5.21.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.5.21.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

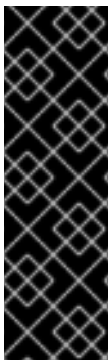
- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	False

23.5.21.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

23.5.21.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

- a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.

- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.5.22. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.12.0	True	False	False

baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

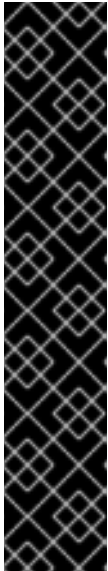
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running   0    5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> ❶
```

- ❶ Specify the pod name and namespace, as shown in the output of the previous command.

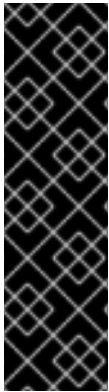
If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

23.5.23. Configuring vSphere DRS anti-affinity rules for control plane nodes

vSphere Distributed Resource Scheduler (DRS) anti-affinity rules can be configured to support higher availability of OpenShift Container Platform Control Plane nodes. Anti-affinity rules ensure that the vSphere Virtual Machines for the OpenShift Container Platform Control Plane nodes are not scheduled to the same vSphere Host.



IMPORTANT

- The following information applies to compute DRS only and does not apply to storage DRS.
- The **govc** command is an open-source command available from VMware; it is not available from Red Hat. The **govc** command is not supported by the Red Hat support.
- Instructions for downloading and installing **govc** are found on the VMware documentation website.

Create an anti-affinity rule by running the following command:

Example command

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

After creating the rule, your control plane nodes are automatically migrated by vSphere so they are not running on the same hosts. This might take some time while vSphere reconciles the new rule. Successful command completion is shown in the following procedure.



NOTE

The migration occurs automatically and might cause brief OpenShift API outage or latency until the migration finishes.

The vSphere DRS anti-affinity rules need to be updated manually in the event of a control plane VM name change or migration to a new vSphere Cluster.

Procedure

1. Remove any existing DRS anti-affinity rule by running the following command:

```
$ govc cluster.rule.remove \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster
```

Example Output

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. Create the rule again with updated names by running the following command:

```
$ govc cluster.rule.create \  
-name openshift4-control-plane-group \  
-dc MyDatacenter -cluster MyOtherCluster \  
-enable \  
-anti-affinity master-0 master-1 master-2
```

23.5.24. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.5.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.5.26. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- [Set up your registry and configure registry storage](#).

- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.6. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure that you provision with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.



NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

23.6.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. Verify that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

23.6.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.6.3. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.54. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.55. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

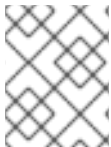
You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.6.4. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.

**NOTE**

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.6.5. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

23.6.5.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 23.16. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.Remove VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 23.17. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine".Change Configuration".Change resource" "Virtual machine".Change Configuration".Change Settings" "Virtual machine".Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine".Edit Inventory".Create new" "Virtual machine".Edit Inventory".Create from existing" "Virtual machine".Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine".Change Configuration".Add existing disk" "Virtual machine".Change Configuration".Add new disk"

vSphere object for role	When required	"Virtual machine". "Change Configuration". "Add or remove device"
		"Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". "Rename" "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. "Reset" "Virtual machine". "Edit Inventory". "Create new"

vSphere object for role	When required	"Virtual machine". "Edit Inventory". "Create from Existing"
		"Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Deploy template" "Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.18. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 23.56. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

- [Creating a compute machine set on vSphere](#)

23.6.5.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 23.57. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

23.6.5.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 23.58. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	Input/Output Per Second (IOPS)[1]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [2]	2	8 GB	100 GB	300

1. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
2. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

23.6.5.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

23.6.5.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow

the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

23.6.5.5.1. Setting the cluster node hostnames through DHCP

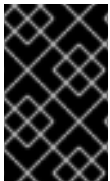
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

23.6.5.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 23.59. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.60. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.61. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00 to 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 to 00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

23.6.5.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.



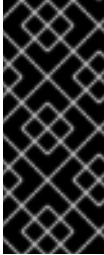
NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.62. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
	api-int.<cluster_name>.<base_domain>.	<p>A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p> </div> </div>
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

23.6.5.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 23.19. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.

- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

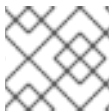
Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 23.20. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

23.6.5.7. Load balancing requirements for user-provisioned infrastructure

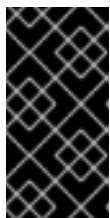
Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

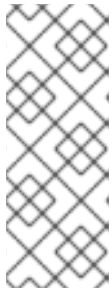
Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 23.63. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer:** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 23.64. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
------	----------------------------------	----------	----------	-------------

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

23.6.5.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 23.21. Sample API and application Ingress load balancer configuration

```
global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request 10s
  timeout queue 1m
  timeout connect 10s
  timeout client 1m
```

```

timeout server      1m
timeout http-keep-alive 10s
timeout check       10s
maxconn             3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:443 check inter 1s
  server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:80 check inter 1s
  server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

23.6.6. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.

**NOTE**

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

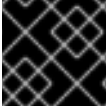
4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

23.6.7. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

- From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

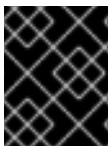
- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

23.6.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.

**IMPORTANT**

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

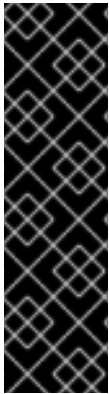
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

23.6.9. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- Failure domain: Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- Region: Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- Zone: Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster than runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.65. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

23.6.10. Obtaining the installation program

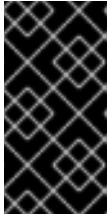
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

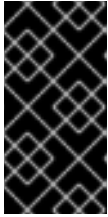
- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

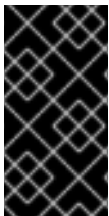
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

23.6.11. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

**IMPORTANT**

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

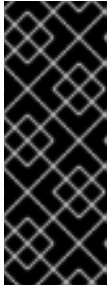
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

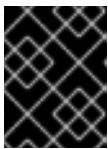
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

23.6.11.1. Sample **install-config.yaml** file for VMware vSphere

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 13
    diskType: thin 14
  fips: false 15
  pullSecret: '{"auths": ...}' 16
  sshKey: 'ssh-ed25519 AAAA...' 17

```


- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, (-), and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 3 You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.
- 5 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 6 The cluster name that you specified in your DNS records.
- 7 The fully-qualified hostname or IP address of the vCenter server.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

- 8 The name of the user for accessing the server.
- 9 The password associated with the vSphere user.
- 10 The vSphere datacenter.
- 11 The default vSphere datastore to use.
- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster and you do not want to use the default **StorageClass** object, named **thin**, you can omit the **folder** parameter from the **install-config.yaml** file.
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.
- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container

enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 The pull secret that you obtained from [OpenShift Cluster Manager Hybrid Cloud Console](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

23.6.11.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

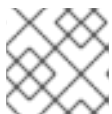
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
```

```

httpProxy: http://<username>:<pswd>@<ip>:<port> 1
httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

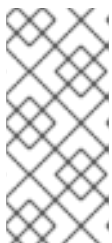
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.6.11.3. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.

**IMPORTANT**

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

**IMPORTANT**

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.

**IMPORTANT**

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.

**NOTE**

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

- To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

- To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

- Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: 2
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: 3
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
metadata:
```

```

name: cluster
platform:
vsphere:
  vcenter: <vcenter_server> 4
  username: <username> 5
  password: <password> 6
  datacenter: datacenter 7
  defaultDatastore: datastore 8
  folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
  cluster: cluster 10
  resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
  diskType: thin
  failureDomains: 12
  - name: <machine_pool_zone_1> 13
    region: <region_tag_1> 14
    zone: <zone_tag_1> 15
    topology: 16
      datacenter: <datacenter1> 17
      computeCluster: "/<datacenter1>/host/<cluster1>" 18
      resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
      networks: 20
      - <VM_Network1_name>
      datastore: "/<datacenter1>/datastore/<datastore1>" 21
  - name: <machine_pool_zone_2>
    region: <region_tag_2>
    zone: <zone_tag_2>
    topology:
      datacenter: <datacenter2>
      computeCluster: "/<datacenter2>/host/<cluster2>"
      networks:
      - <VM_Network2_name>
      datastore: "/<datacenter2>/datastore/<datastore2>"
      resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
      folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.

- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.6.12. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

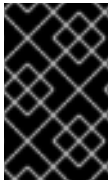
Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

23.6.13. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

- Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

- Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

- Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider


```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800

```

Enable IPsec for the OVN-Kubernetes network provider

```

apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}

```

- Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.
- Remove the Kubernetes manifest files that define the control plane machines and compute machineSets:

```

$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml

```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the MachineSet files to create compute machines by using the machine API, but you must update references to them to match your environment.

23.6.14. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

23.6.14.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 23.66. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 23.67. defaultNetwork object

Field	Type	Description
-------	------	-------------

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 23.68. openshiftSDNConfig object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 23.69. `ovnKubernetesConfig` object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 23.70. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 23.71. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>

Example OVN-Kubernetes configuration with IPsec enabled

```


defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}

```

kubeProxyConfig object configuration

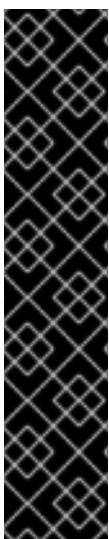
The values for the **kubeProxyConfig** object are defined in the following table:

Table 23.72. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

23.6.15. Creating the Ignition config files

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

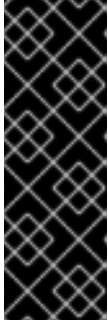
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

- Obtain the Ignition config files:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.



IMPORTANT

If you created an **install-config.yaml** file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

The following files are generated in the directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.6.16. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

23.6.17. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {},
```

```
"storage": {},
"systemd": {}
}
```

- 1 Specify the URL of the bootstrap Ignition config file that you hosted.

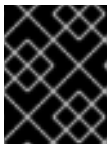
When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:
 - **<installation_directory>/master.ign**
 - **<installation_directory>/worker.ign**
 - **<installation_directory>/merge-bootstrap.ign**
4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

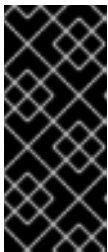
```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
 - a. Click the **VMs and Templates** view.
 - b. Right-click the name of your datacenter.

- c. Click **New Folder → New VM and Template Folder**.
 - d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.



NOTE

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

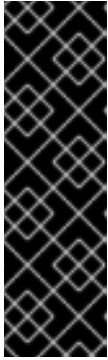
- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
 - Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
- g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.



IMPORTANT

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.



IMPORTANT

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone → Clone to Virtual Machine**
 - b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.



IMPORTANT

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```

- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

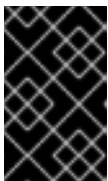
- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.
- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

23.6.18. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone → Clone to Virtual Machine**
2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**
7. On the **Customize hardware** tab, click **Advanced**.
 - a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

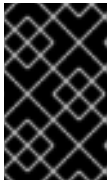
- Continue to create more compute machines for your cluster.

23.6.19. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions:** For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.



IMPORTANT

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

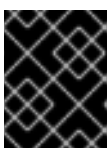
- **Retain existing partitions:** For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate **/var** partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers:** Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd:** Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var:** Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this

method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

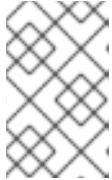
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.

- 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

23.6.20. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

23.6.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

23.6.22. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.25.0
master-1	Ready	master	63m	v1.25.0
master-2	Ready	master	64m	v1.25.0

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-8b2br	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
csr-8vnps	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
...			

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-bfd72	5m26s	system:node:ip-10-0-50-126.us-east-2.compute.internal	Pending
csr-c57lv	5m26s	system:node:ip-10-0-95-157.us-east-2.compute.internal	Pending
...			

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	73m	v1.25.0
master-1	Ready	master	73m	v1.25.0
master-2	Ready	master	74m	v1.25.0
worker-0	Ready	worker	11m	v1.25.0
worker-1	Ready	worker	11m	v1.25.0



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

23.6.22.1. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

23.6.22.2. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

23.6.22.3. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.6.22.3.1. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
    - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.

4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.6.23. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m

dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

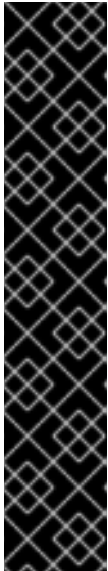
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running   0    5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

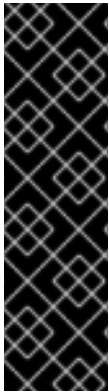
If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

23.6.24. Configuring vSphere DRS anti-affinity rules for control plane nodes

vSphere Distributed Resource Scheduler (DRS) anti-affinity rules can be configured to support higher availability of OpenShift Container Platform Control Plane nodes. Anti-affinity rules ensure that the vSphere Virtual Machines for the OpenShift Container Platform Control Plane nodes are not scheduled to the same vSphere Host.



IMPORTANT

- The following information applies to compute DRS only and does not apply to storage DRS.
- The **govc** command is an open-source command available from VMware; it is not available from Red Hat. The **govc** command is not supported by the Red Hat support.
- Instructions for downloading and installing **govc** are found on the VMware documentation website.

Create an anti-affinity rule by running the following command:

Example command

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

After creating the rule, your control plane nodes are automatically migrated by vSphere so they are not running on the same hosts. This might take some time while vSphere reconciles the new rule. Successful command completion is shown in the following procedure.



NOTE

The migration occurs automatically and might cause brief OpenShift API outage or latency until the migration finishes.

The vSphere DRS anti-affinity rules need to be updated manually in the event of a control plane VM name change or migration to a new vSphere Cluster.

Procedure

1. Remove any existing DRS anti-affinity rule by running the following command:

```
$ govc cluster.rule.remove \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster
```

Example Output

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. Create the rule again with updated names by running the following command:

```
$ govc cluster.rule.create \  
-name openshift4-control-plane-group \  
-dc MyDatacenter -cluster MyOtherCluster \  
-enable \  
-anti-affinity master-0 master-1 master-2
```

23.6.25. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.6.26. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.6.27. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- [Set up your registry and configure registry storage](#).

- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.7. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.12, you can install a cluster on VMware vSphere infrastructure in a restricted network by creating an internal mirror of the installation release content.

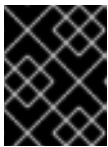


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

23.7.1. Prerequisites

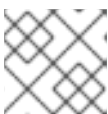
- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide the ReadWriteMany access mode.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

If you are configuring a proxy, be sure to also review this site list.

23.7.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on

the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

23.7.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

23.7.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.7.4. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.73. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

**IMPORTANT**

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.74. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.7.5. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 23.75. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.76. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.77. Ports used for control plane machine to control plane machine communications

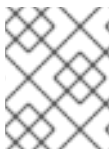
Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

23.7.6. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.7.7. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder

to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 23.22. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	

vSphere object for role	When required	InventoryService.Tagging.O bject.Attachable Resource.AssignVMToPool API VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk VirtualMachine.Config.Rena me VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone VirtualMachine.Provisionin g.MarkAsTemplate VirtualMachine.Provisionin g.DeployTemplate

vSphere object for role	When required	Required privileges in vSphere
	<p>The installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.</p>	<p>APIAttachableResource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning</p>

vSphere object for role	When required	Required privileges in vSphere API
		g.MarkAsTemplate Power.Create Folder.Delete

Example 23.23. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration". "Add new disk"
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration". "Add existing disk" "Virtual machine"."Change Configuration". "Add new disk" "Virtual machine"."Change Configuration". "Add or remove device" "Virtual machine"."Change Configuration". "Advanced configuration" "Virtual machine"."Change Configuration". "Set annotation" "Virtual machine"."Change Configuration". "Change CPU count" "Virtual machine"."Change Configuration". "Extend virtual disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Acquire Required privileges in vCenter GUI "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit Inventory". "Create from existing" "Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Mark as template" "Virtual machine". Provisioning. "Deploy template"
vSphere vCenter Datacenter	If the installation program	"vSphere Tagging". "Assign

vSphere object for role	<p>creates the virtual machine folder, CPU, and</p> <p>When required</p> <p>VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.</p>	<p>or Unassign vSphere Tag on Object</p> <p>Required privileges in vCenter GUI</p> <p>"Resource."Assign virtual machine to resource pool"</p>
		<p>VApp.Import</p> <p>"Virtual machine"."Change Configuration"."Add existing disk"</p> <p>"Virtual machine"."Change Configuration"."Add new disk"</p> <p>"Virtual machine"."Change Configuration"."Add or remove device"</p> <p>"Virtual machine"."Change Configuration"."Advanced configuration"</p> <p>"Virtual machine"."Change Configuration"."Set annotation"</p> <p>"Virtual machine"."Change Configuration"."Change CPU count"</p> <p>"Virtual machine"."Change Configuration"."Extend virtual disk"</p> <p>"Virtual machine"."Change Configuration"."Acquire disk lease"</p> <p>"Virtual machine"."Change Configuration"."Modify device settings"</p> <p>"Virtual machine"."Change Configuration"."Change Memory"</p> <p>"Virtual machine"."Change Configuration"."Remove disk"</p> <p>"Virtual machine"."Change Configuration".Rename</p> <p>"Virtual machine"."Change Configuration"."Reset guest information"</p> <p>"Virtual machine"."Change Configuration"."Change resource"</p> <p>"Virtual machine"."Change Configuration"."Change Settings"</p> <p>"Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility"</p> <p>"Virtual machine".Interaction."Guest operating system"</p>

vSphere object for role	When required	management by VIX API" Required privileges in vCenter GUI "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."De ploy template" "Virtual machine".Provisioning."Mar k as template" Folder."Create folder" Folder."Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.24. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category

- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. The VM in your restricted network must have access to vCenter so that it can provision and manage nodes, persistent volume claims (PVCs), and other resources. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 23.78. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

23.7.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **`./openshift-install gather`** command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

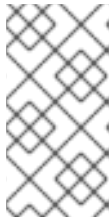
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

23.7.9. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl
```

3 directories, 15 files

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:


```
# update-ca-trust extract
```

23.7.10. Creating the RHCOS image for restricted network installations

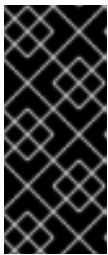
Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network VMware vSphere environment.

Prerequisites

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

Procedure

1. Log in to the Red Hat Customer Portal's [Product Downloads](#) page .
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.12 for RHEL 8.



IMPORTANT

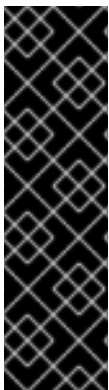
The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - vSphere** image.
4. Upload the image you downloaded to a location that is accessible from the bastion server.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

23.7.11. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#) .

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.79. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a

Datacenter (region)	Cluster (zone)	Tags
		us-west-2b

23.7.12. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
 - iii. Specify the name of your vCenter instance.
 - iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
 - v. Select the data center in your vCenter instance to connect to.
 - vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. In the **install-config.yaml** file, set the value of **platform.vsphere.clusterOSImage** to the image location or name. For example:

```
platform:
  vsphere:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-vmware.x86_64.ova?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email": "you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----/
  -----END CERTIFICATE-----
```

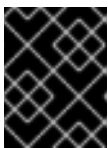
The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

4. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

23.7.12.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

23.7.12.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 23.80. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

23.7.12.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 23.81. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>

Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

23.7.12.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 23.82. Optional parameters

Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master

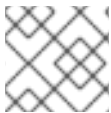
Parameter	Description	Values
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code>
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 20px; height: 20px; border: 1px solid black; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 20px; height: 20px; border: 1px solid black; margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div> </div>	Mint, Passthrough, Manual or an empty string (<code>""</code>).

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 584 592 1391" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: #f0f0f0; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true

Parameter	Description	Values
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 40px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 40px; height: 40px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA...

23.7.12.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table.





NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

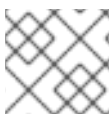
Table 23.83. Additional VMware vSphere cluster parameters

Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String

Parameter	Description	Values
apiVIPs	<p>The virtual IP (VIP) address that you configured for control plane API access.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

23.7.12.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.84. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware-<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer

Parameter	Description	Values
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

23.7.12.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 23.85. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String

Parameter	Description	Values
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String


```

-----END CERTIFICATE-----
imageContentSources: 13
- mirrors:
- <mirror_host_name>:<mirror_port>/<repo_name>/release
  source: <source_image_1>
- mirrors:
- <mirror_host_name>:<mirror_port>/<repo_name>/release-images
  source: <source_image_2>

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
- 6 The cluster name that you specified in your DNS records.
- 7 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
- 8 The vSphere disk provisioning method.
- 9 The vSphere cluster to install the OpenShift Container Platform cluster in.
- 10 The location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that is accessible from the bastion server.
- 11 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 12 Provide the contents of the certificate file that you used for your mirror registry.
- 13 Provide the **imageContentSources** section from the output of the command to mirror the repository.



NOTE

In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

23.7.12.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

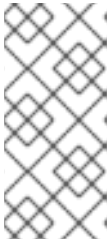
- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when

Always. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

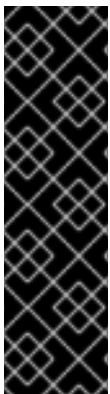


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.7.12.4. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).



IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

- Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

- To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

- To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

- Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```

apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
    datacenter: datacenter ❼
    defaultDatastore: datastore ❽
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" ❾
    cluster: cluster ❿
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" ❶❶
    diskType: thin
    failureDomains: ❶❷
      - name: <machine_pool_zone_1> ❶❸
        region: <region_tag_1> ❶❹
        zone: <zone_tag_1> ❶❺
        topology: ❶❻
          datacenter: <datacenter1> ❶❼
          computeCluster: "/<datacenter1>/host/<cluster1>" ❶❶❶
          resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" ❶❶❷
          networks: ❶❶❸
            - <VM_Network1_name>
          datastore: "/<datacenter1>/datastore/<datastore1>" ❶❶❹
      - name: <machine_pool_zone_2>
        region: <region_tag_2>
        zone: <zone_tag_2>
        topology:
          datacenter: <datacenter2>
          computeCluster: "/<datacenter2>/host/<cluster2>"
          networks:

```

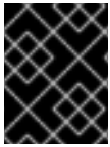
```
- <VM_Network2_name>
  datastore: "/<datacenter2>/datastore/<datastore2>"
  resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
  folder: "/<datacenter2>/vm/<folder2>"
```

...

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.7.13. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
```

```

INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s

```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

23.7.14. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

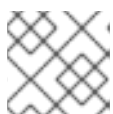
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

23.7.15. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

23.7.16. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \  
-p [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

23.7.17. Creating registry storage

After you install the cluster, you must create storage for the Registry Operator.

23.7.17.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

23.7.17.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.7.17.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

-

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

23.7.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

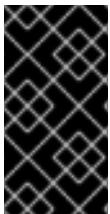
After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.7.19. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 23.7. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment

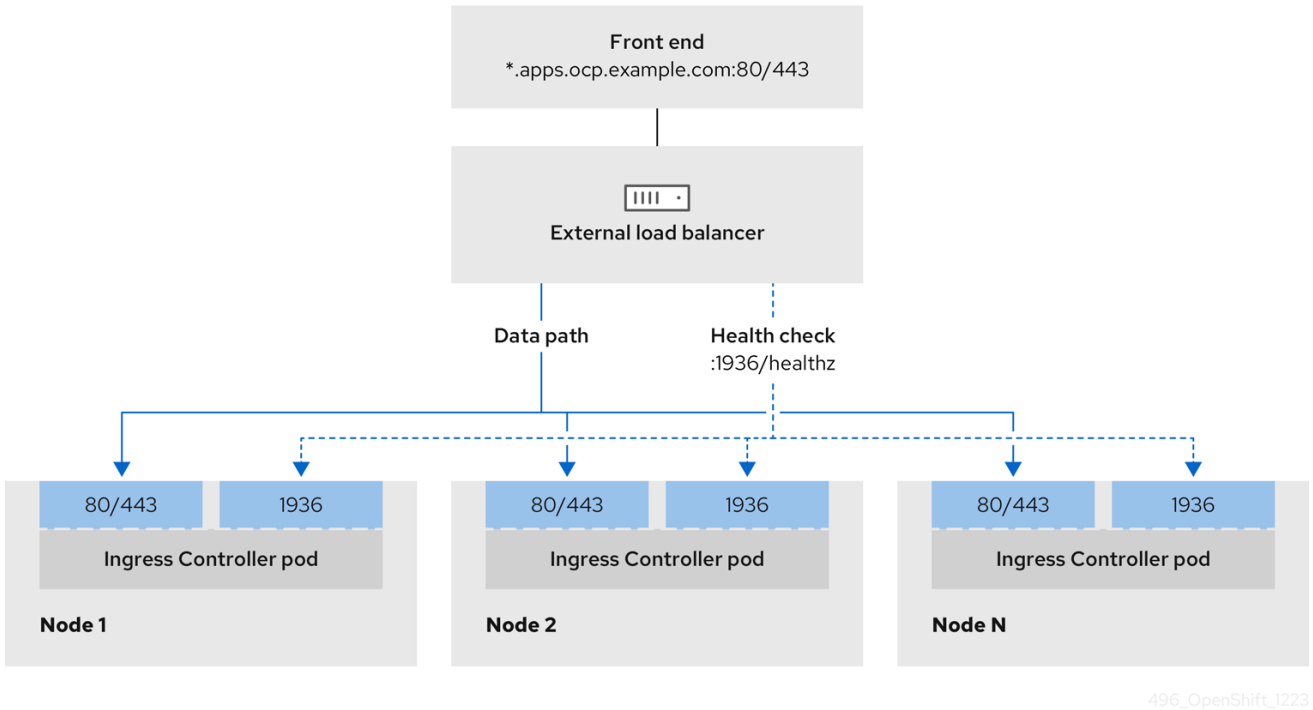


Figure 23.8. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment

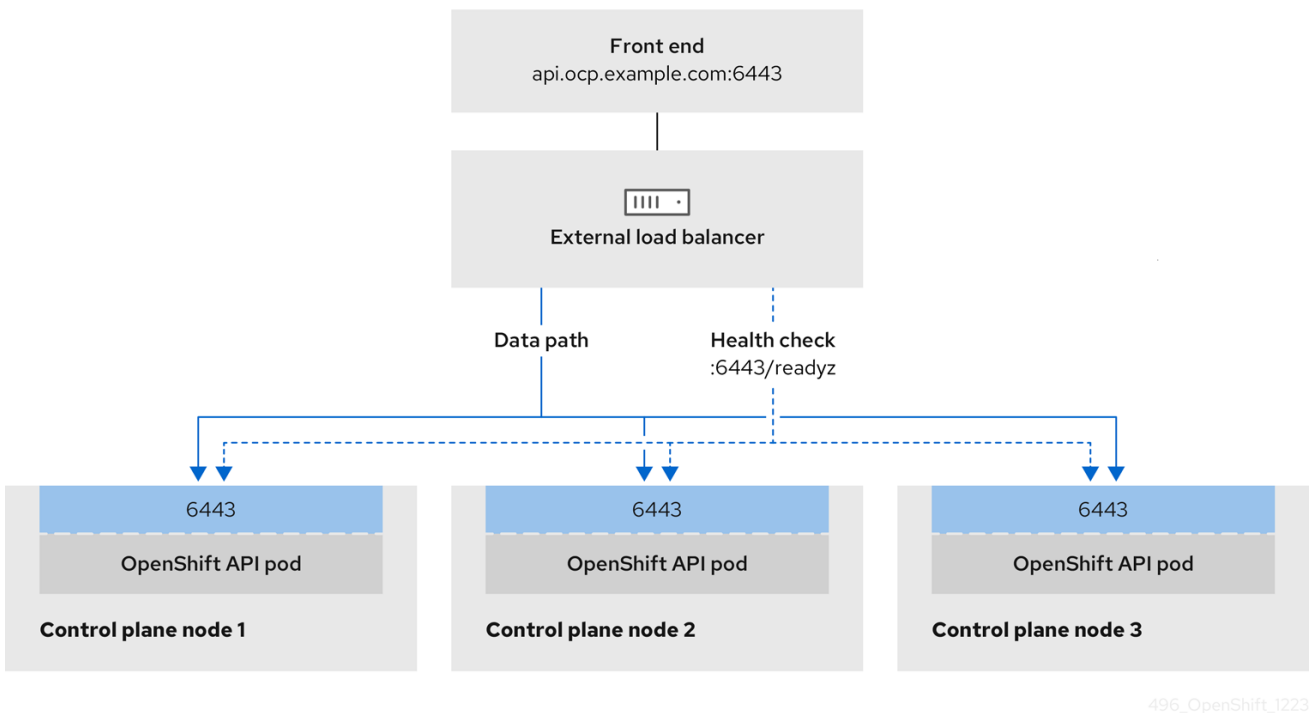
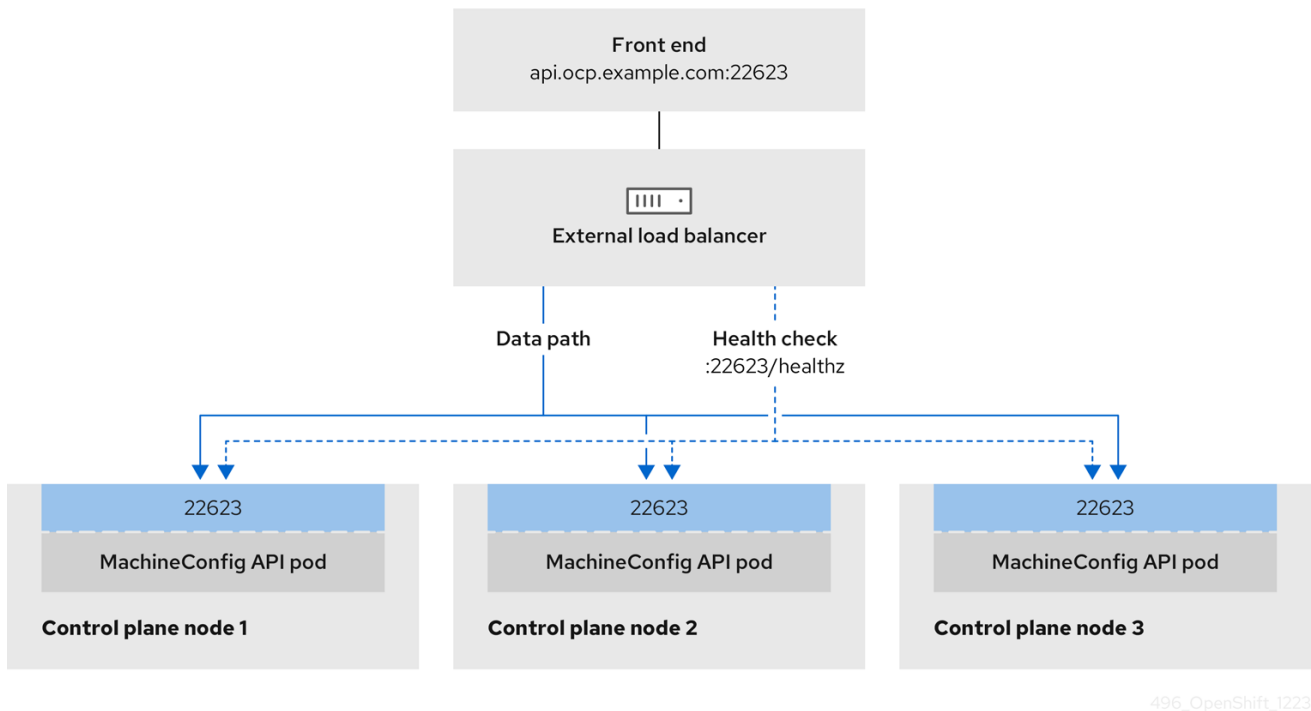


Figure 23.9. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a /27 or /28, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

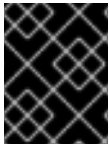
Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

23.7.19.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
}
```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie: 1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/; HttpOnly; Secure; SameSite=None
cache-control: private

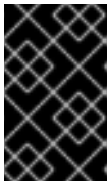
```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

23.7.20. Next steps

- [Customize your cluster.](#)

- If necessary, you can [opt out of remote health reporting](#) .
- If necessary, see [Registering your disconnected cluster](#)
- [Set up your registry and configure registry storage](#) .

23.8. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure that you provision in a restricted network.



NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

23.8.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [persistent storage](#) for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.
- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.

- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

23.8.2. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

23.8.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

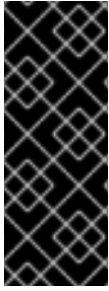
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

23.8.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

23.8.4. VMware vSphere infrastructure requirements

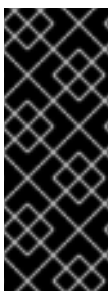
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 23.86. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

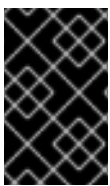
**IMPORTANT**

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 23.87. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
-----------	----------------------------	-------------

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.
Optional: Networking (NSX-T)	vSphere 7.0 Update 2 or later; vSphere 8.0 Update 1	At a minimum, vSphere 7.0 Update 2 or vSphere 8.0 Update 1 is required for OpenShift Container Platform. For more information about the compatibility of NSX and OpenShift Container Platform, see the Release Notes section of VMware's NSX container plugin documentation .



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

23.8.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.

**NOTE**

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party vSphere CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

23.8.6. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

23.8.6.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 23.25. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk

vSphere object for role	When required	VirtualMachine.Config.Rena Required privileges in vSphere API VirtualMachine.Config.Rese tGuestInfo
		VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.Rem

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 23.26. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration".Change resource" "Virtual machine"."Change Configuration".Change Settings" "Virtual machine"."Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory".Create new" "Virtual machine"."Edit Inventory".Create from existing" "Virtual machine"."Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration".Add existing disk" "Virtual machine"."Change Configuration".Add new disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Add or Remove device Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from existing virtual machine". "Edit virtual machine". "Remove virtual machine". "Clone virtual machine". "Deploy template". "Mark as template". "Create folder". "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 23.27. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you

require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.88. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>..	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>..	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

- [Creating a compute machine set on vSphere](#)

23.8.6.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 23.89. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

23.8.6.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 23.90. Minimum resource requirements

Machine	Operating System	vCPU	Virtual RAM	Storage	Input/Output Per Second (IOPS)[1]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [2]	2	8 GB	100 GB	300

1. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so

you might need to over-allocate storage volume to obtain sufficient performance.

2. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- [Optimizing storage](#)

23.8.6.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

23.8.6.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

23.8.6.5.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

23.8.6.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 23.91. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 23.92. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 23.93. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00 to 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 to 00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

23.8.6.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 23.94. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <h4>IMPORTANT</h4> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

23.8.6.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 23.28. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

- 1** Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2** Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3** Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4** Provides name resolution for the bootstrap machine.
- 5 6 7** Provides name resolution for the control plane machines.
- 8 9** Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 23.29. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1** Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2** Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3** Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6** Provides reverse DNS resolution for the control plane machines.

7 8 Provides reverse DNS resolution for the compute machines.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard.

23.8.6.7. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

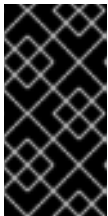


NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



IMPORTANT

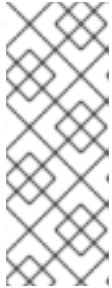
Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 23.95. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server

Port	Back-end machines (pool members)	Internal	External	Description
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 23.96. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

23.8.6.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an `/etc/haproxy/haproxy.cfg` configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 23.30. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile /var/run/haproxy.pid
  maxconn 4000
  daemon
defaults
  mode          http
  log           global
  option        dontlognull
  option http-server-close
  option        redispatch
  retries       3
  timeout http-request 10s
  timeout queue 1m
  timeout connect 10s
  timeout client 1m
  timeout server 1m
  timeout http-keep-alive 10s
  timeout check 10s
  maxconn      3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s

```

```

fall 2 rise 3
server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 3
bind *:22623
mode tcp
server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
server master0 master0.ocp4.example.com:22623 check inter 1s
server master1 master1.ocp4.example.com:22623 check inter 1s
server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1 Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2 4 The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3 Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5 Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6 Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

23.8.7. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

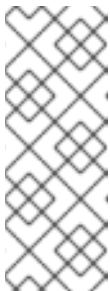
After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.

3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.
 - a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
 - a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.



NOTE

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

23.8.8. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.



IMPORTANT

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

- From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1  
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

23.8.9. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

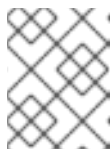
On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

23.8.10. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vccenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 23.97. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

23.8.11. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain the **imageContentSources** section from the output of the command to mirror the repository.
- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

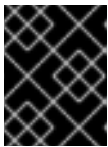
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

**NOTE**

You must name this configuration file **install-config.yaml**.

- Unless you use a registry that RHCOS trusts by default, such as **docker.io**, you must provide the contents of the certificate for your mirror repository in the **additionalTrustBundle** section. In most cases, you must provide the certificate for your mirror.
 - You must include the **imageContentSources** section from the output of the command to mirror the repository.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

**IMPORTANT**

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

23.8.11.1. Sample **install-config.yaml** file for VMware vSphere

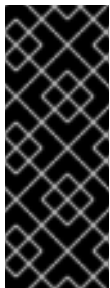
You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12

```


- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example,
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.
- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.
- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 18 Provide the contents of the certificate file that you used for your mirror registry.
- 19 Provide the **imageContentSources** section from the output of the command to mirror the repository.

23.8.11.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

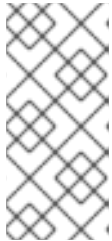
- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **AlwaysUseProxyonly** to reference the **user-ca-bundle** config map only when

Always. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



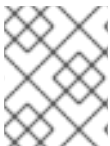
NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

23.8.11.3. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

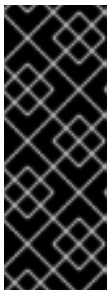


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.

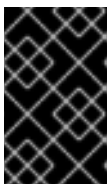


NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```


- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```

apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
    datacenter: datacenter ❼
    defaultDatastore: datastore ❽
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" ❾
    cluster: cluster ❿
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" ❶❶
    diskType: thin
    failureDomains: ❶❷
    - name: <machine_pool_zone_1> ❶❸
      region: <region_tag_1> ❶❹
      zone: <zone_tag_1> ❶❺
    topology: ❶❻
      datacenter: <datacenter1> ❶❼
      computeCluster: "/<datacenter1>/host/<cluster1>" ❶❶❶
      resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" ❶❶❷
      networks: ❶❶❸
      - <VM_Network1_name>
      datastore: "/<datacenter1>/datastore/<datastore1>" ❶❶❹
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
    topology:
      datacenter: <datacenter2>
      computeCluster: "/<datacenter2>/host/<cluster2>"
      networks:

```

```
- <VM_Network2_name>
  datastore: "/<datacenter2>/datastore/<datastore2>"
  resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
  folder: "/<datacenter2>/vm/<folder2>"
```

...

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

23.8.12. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
3. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane

machines:

- a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
 - b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
 - c. Save and exit the file.
4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> ❶
```

- ❶ For `<installation_directory>`, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The `kubeadmin-password` and `kubeconfig` files are created in the `./<installation_directory>/auth` directory:

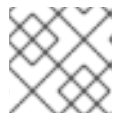
```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.8.13. Configuring chrony time service

You must set the time server and related settings used by the chrony time service (`chronyd`) by modifying the contents of the `chrony.conf` file and passing those contents to your nodes as a machine config.

Procedure

1. Create a Butane config including the contents of the `chrony.conf` file. For example, to configure chrony on worker nodes, create a `99-worker-chrony.bu` file.



NOTE

See "Creating machine configs with Butane" for information about Butane.

```
variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-chrony ❶
  labels:
    machineconfiguration.openshift.io/role: worker ❷
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644 ❸
```

```

overwrite: true
contents:
  inline: |
    pool 0.rhel.pool.ntp.org iburst 4
    driftfile /var/lib/chrony/drift
    makestep 1.0 3
    rtsync
    logdir /var/log/chrony

```

- 1 2 On control plane nodes, substitute **master** for **worker** in both of these locations.
 - 3 Specify an octal value mode for the **mode** field in the machine config file. After creating the file and applying the changes, the **mode** is converted to a decimal value. You can check the YAML file with the command **oc get mc <mc-name> -o yaml**.
 - 4 Specify any valid, reachable time source, such as the one provided by your DHCP server.
2. Use Butane to generate a **MachineConfig** object file, **99-worker-chrony.yaml**, containing the configuration to be delivered to the nodes:

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. Apply the configurations in one of two ways:
 - If the cluster is not running yet, after you generate manifest files, add the **MachineConfig** object file to the **<installation_directory>/openshift** directory, and then continue to create the cluster.
 - If the cluster is already running, apply the file:

```
$ oc apply -f ./99-worker-chrony.yaml
```

23.8.14. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

23.8.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {}
}
```

```
"storage": {},
"systemd": {}
}
```

- 1 Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

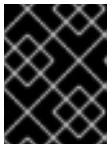
- **<installation_directory>/master.ign**
- **<installation_directory>/worker.ign**
- **<installation_directory>/merge-bootstrap.ign**

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

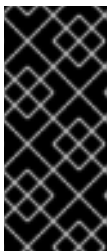
```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
 - a. Click the **VMs and Templates** view.
 - b. Right-click the name of your datacenter.

- c. Click **New Folder → New VM and Template Folder**.
 - d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.



NOTE

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

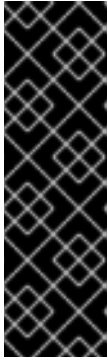
- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
 - Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
- g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.



IMPORTANT

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.



IMPORTANT

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone → Clone to Virtual Machine**
 - b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.



IMPORTANT

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```

- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

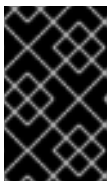
- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.
- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

23.8.16. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

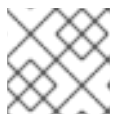
After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone → Clone to Virtual Machine**
2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**
7. On the **Customize hardware** tab, click **Advanced**.
 - a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

- Continue to create more compute machines for your cluster.

23.8.17. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

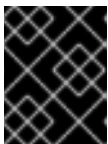
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.



IMPORTANT

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate **/var** partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this

method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

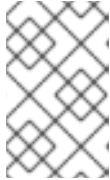
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.

- 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

23.8.18. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

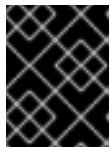
2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

23.8.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

-

Example output

```
system:admin
```

23.8.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   63m   v1.25.0
master-1  Ready   master   63m   v1.25.0
master-2  Ready   master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE    REQUESTOR                                     CONDITION
csr-bfd72 5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
```

```
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

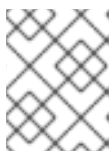
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}
{{end}}{{end}} | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

23.8.21. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

23.8.21.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

23.8.21.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

23.8.21.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	False

23.8.21.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

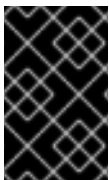
If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

23.8.21.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

- Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy":"Recreate","replicas":1}}'
```

- Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
```

```

name: image-registry-storage 1
namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
  requests:
    storage: 100Gi 4

```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```

storage:
  pvc:
    claim: 1

```

- 1** By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

23.8.22. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

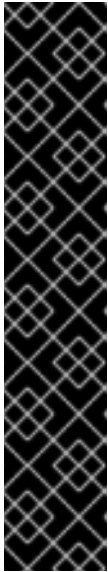
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```


The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS   RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1    Running   0          9m
openshift-apiserver          apiserver-67b9g                                1/1    Running   0          3m
openshift-apiserver          apiserver-ljcmx                                1/1    Running   0          1m
openshift-apiserver          apiserver-z25h4                                1/1    Running   0          2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1    Running   0          5m
...

```

b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.
4. Register your cluster on the [Cluster registration](#) page.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

23.8.23. Configuring vSphere DRS anti-affinity rules for control plane nodes

vSphere Distributed Resource Scheduler (DRS) anti-affinity rules can be configured to support higher availability of OpenShift Container Platform Control Plane nodes. Anti-affinity rules ensure that the vSphere Virtual Machines for the OpenShift Container Platform Control Plane nodes are not scheduled to the same vSphere Host.



IMPORTANT

- The following information applies to compute DRS only and does not apply to storage DRS.
- The **govc** command is an open-source command available from VMware; it is not available from Red Hat. The **govc** command is not supported by the Red Hat support.
- Instructions for downloading and installing **govc** are found on the VMware documentation website.

Create an anti-affinity rule by running the following command:

Example command

```
$ govc cluster.rule.create \  
-name openshift4-control-plane-group \  
-dc MyDatacenter -cluster MyCluster \  
-enable \  
-anti-affinity master-0 master-1 master-2
```

After creating the rule, your control plane nodes are automatically migrated by vSphere so they are not running on the same hosts. This might take some time while vSphere reconciles the new rule. Successful command completion is shown in the following procedure.



NOTE

The migration occurs automatically and might cause brief OpenShift API outage or latency until the migration finishes.

The vSphere DRS anti-affinity rules need to be updated manually in the event of a control plane VM name change or migration to a new vSphere Cluster.

Procedure

1. Remove any existing DRS anti-affinity rule by running the following command:

```
$ govc cluster.rule.remove \
-name openshift4-control-plane-group \
-dc MyDatacenter -cluster MyCluster
```

Example Output

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. Create the rule again with updated names by running the following command:

```
$ govc cluster.rule.create \
-name openshift4-control-plane-group \
-dc MyDatacenter -cluster MyOtherCluster \
-enable \
-anti-affinity master-0 master-1 master-2
```

23.8.24. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

23.8.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

23.8.26. Next steps

- [Customize your cluster](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

23.9. CONFIGURING THE VSPHERE CONNECTION SETTINGS AFTER AN INSTALLATION

After installing an OpenShift Container Platform cluster on vSphere with the platform integration feature enabled, you might need to update the vSphere connection settings manually, depending on the installation method.

For installations using the Assisted Installer, you must update the connection settings. This is because the Assisted Installer adds default connection settings to the **vSphere connection configuration** wizard as placeholders during the installation.

For installer-provisioned or user-provisioned infrastructure installations, you should have entered valid connection settings during the installation. You can use the **vSphere connection configuration** wizard at any time to validate or modify the connection settings, but this is not mandatory for completing the installation.

23.9.1. Configuring the vSphere connection settings

Modify the following vSphere configuration settings as required:

- vCenter address
- vCenter cluster
- vCenter username
- vCenter password
- vCenter address
- vSphere data center
- vSphere datastore
- Virtual machine folder

Prerequisites

- The Assisted Installer has finished installing the cluster successfully.
- The cluster is connected to <https://console.redhat.com>.

Procedure

1. In the Administrator perspective, navigate to **Home → Overview**.

2. Under **Status**, click **vSphere connection** to open the **vSphere connection configuration** wizard.
3. In the **vCenter** field, enter the network address of the vSphere vCenter server. This can be either a domain name or an IP address. It appears in the vSphere web client URL; for example **https://[your_vCenter_address]/ui**.
4. In the **vCenter cluster** field, enter the name of the vSphere vCenter cluster where OpenShift Container Platform is installed.



IMPORTANT

This step is mandatory if you installed OpenShift Container Platform 4.13 or later.

5. In the **Username** field, enter your vSphere vCenter username.
6. In the **Password** field, enter your vSphere vCenter password.



WARNING

The system stores the username and password in the **vsphere-creds** secret in the **kube-system** namespace of the cluster. An incorrect vCenter username or password makes the cluster nodes unschedulable.

7. In the **Datacenter** field, enter the name of the vSphere data center that contains the virtual machines used to host the cluster; for example, **SDDC-Datacenter**.
8. In the **Default data store** field, enter the path and name of the vSphere data store that stores the persistent data volumes; for example, **/SDDC-Datacenter/datastore/datastorename**.



WARNING

Updating the vSphere data center or default data store after the configuration has been saved detaches any active vSphere **PersistentVolumes**.

9. In the **Virtual Machine Folder** field, enter the data center folder that contains the virtual machine of the cluster; for example, **/SDDC-Datacenter/vm/ci-ln-hjg4vg2-c61657-t2gzs**. For the OpenShift Container Platform installation to succeed, all virtual machines comprising the cluster must be located in a single data center folder.
10. Click **Save Configuration**. This updates the **cloud-provider-config** ConfigMap resource in the **openshift-config** namespace, and starts the configuration process.

11. Reopen the **vSphere connection configuration** wizard and expand the **Monitored operators** panel. Check that the status of the operators is either **Progressing** or **Healthy**.

23.9.2. Verifying the configuration

The connection configuration process updates operator statuses and control plane nodes. It takes approximately an hour to complete. During the configuration process, the nodes will reboot. Previously bound **PersistentVolumeClaims** objects might become disconnected.

Prerequisites

- You have saved the configuration settings in the **vSphere connection configuration** wizard.

Procedure

1. Check that the configuration process completed successfully:
 - a. In the OpenShift Container Platform Administrator perspective, navigate to **Home → Overview**.
 - b. Under **Status** click **Operators**. Wait for all operator statuses to change from **Progressing** to **All succeeded**. A **Failed** status indicates that the configuration failed.
 - c. Under **Status**, click **Control Plane**. Wait for the response rate of all Control Plane components to return to 100%. A **Failed** control plane component indicates that the configuration failed.

A failure indicates that at least one of the connection settings is incorrect. Change the settings in the **vSphere connection configuration** wizard and save the configuration again.

2. Check that you are able to bind **PersistentVolumeClaims** objects by performing the following steps:
 - a. Create a **StorageClass** object using the following YAML:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: vsphere-sc
provisioner: kubernetes.io/vsphere-volume
parameters:
  datastore: YOURVCENTERDATASTORE
  diskformat: thin
  reclaimPolicy: Delete
  volumeBindingMode: Immediate
```

- b. Create a **PersistentVolumeClaims** object using the following YAML:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc
  namespace: openshift-config
annotations:
  volume.beta.kubernetes.io/storage-provisioner: kubernetes.io/vsphere-volume
finalizers:
```

```

- kubernetes.io/pvc-protection
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: vsphere-sc
  volumeMode: Filesystem

```

If you are unable to create a **PersistentVolumeClaims** object, you can troubleshoot by navigating to **Storage** → **PersistentVolumeClaims** in the **Administrator** perspective of the OpenShift Container Platform web console.

For instructions on creating storage objects, see [Dynamic provisioning](#).

23.10. UNINSTALLING A CLUSTER ON VSPHERE THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE

You can remove a cluster that you deployed in your VMware vSphere instance by using installer-provisioned infrastructure.



NOTE

When you run the **openshift-install destroy cluster** command to uninstall OpenShift Container Platform, vSphere volumes are not automatically deleted. The cluster administrator must manually find the vSphere volumes and delete them.

23.10.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```

$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2

```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

23.11. USING THE VSPHERE PROBLEM DETECTOR OPERATOR

23.11.1. About the vSphere Problem Detector Operator

The vSphere Problem Detector Operator checks clusters that are deployed on vSphere for common installation and misconfiguration issues that are related to storage.

The Operator runs in the **openshift-cluster-storage-operator** namespace and is started by the Cluster Storage Operator when the Cluster Storage Operator detects that the cluster is deployed on vSphere. The vSphere Problem Detector Operator communicates with the vSphere vCenter Server to determine the virtual machines in the cluster, the default datastore, and other information about the vSphere vCenter Server configuration. The Operator uses the credentials from the Cloud Credential Operator to connect to vSphere.

The Operator runs the checks according to the following schedule:

- The checks run every 8 hours.
- If any check fails, the Operator runs the checks again in intervals of 1 minute, 2 minutes, 4, 8, and so on. The Operator doubles the interval up to a maximum interval of 8 hours.
- When all checks pass, the schedule returns to an 8 hour interval.

The Operator increases the frequency of the checks after a failure so that the Operator can report success quickly after the failure condition is remedied. You can run the Operator manually for immediate troubleshooting information.

23.11.2. Running the vSphere Problem Detector Operator checks

You can override the schedule for running the vSphere Problem Detector Operator checks and run the checks immediately.

The vSphere Problem Detector Operator automatically runs the checks every 8 hours. However, when the Operator starts, it runs the checks immediately. The Operator is started by the Cluster Storage Operator when the Cluster Storage Operator starts and determines that the cluster is running on vSphere. To run the checks immediately, you can scale the vSphere Problem Detector Operator to **0** and back to **1** so that it restarts the vSphere Problem Detector Operator.

Prerequisites

- Access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Scale the Operator to **0**:

```
$ oc scale deployment/vsphere-problem-detector-operator --replicas=0 \
-n openshift-cluster-storage-operator
```

If the deployment does not scale to zero immediately, you can run the following command to wait for the pods to exit:

```
$ oc wait pods -l name=vsphere-problem-detector-operator \
--for=delete --timeout=5m -n openshift-cluster-storage-operator
```

2. Scale the Operator back to **1**:

```
$ oc scale deployment/vsphere-problem-detector-operator --replicas=1 \
-n openshift-cluster-storage-operator
```

3. Delete the old leader lock to speed up the new leader election for the Cluster Storage Operator:

```
$ oc delete -n openshift-cluster-storage-operator \
cm vsphere-problem-detector-lock
```

Verification

- View the events or logs that are generated by the vSphere Problem Detector Operator. Confirm that the events or logs have recent timestamps.

23.11.3. Viewing the events from the vSphere Problem Detector Operator

After the vSphere Problem Detector Operator runs and performs the configuration checks, it creates events that can be viewed from the command line or from the OpenShift Container Platform web console.

Procedure

- To view the events by using the command line, run the following command:

```
$ oc get event -n openshift-cluster-storage-operator \
--sort-by={.metadata.creationTimestamp}
```

Example output

```
16m Normal Started pod/vsphere-problem-detector-operator-xxxxx Started
container vsphere-problem-detector
16m Normal Created pod/vsphere-problem-detector-operator-xxxxx Created
container vsphere-problem-detector
16m Normal LeaderElection configmap/vsphere-problem-detector-lock vsphere-
problem-detector-operator-xxxxx became leader
```

- To view the events by using the OpenShift Container Platform web console, navigate to **Home** → **Events** and select **openshift-cluster-storage-operator** from the **Project** menu.

23.11.4. Viewing the logs from the vSphere Problem Detector Operator

After the vSphere Problem Detector Operator runs and performs the configuration checks, it creates log records that can be viewed from the command line or from the OpenShift Container Platform web console.

Procedure

- To view the logs by using the command line, run the following command:

```
$ oc logs deployment/vsphere-problem-detector-operator \
-n openshift-cluster-storage-operator
```

Example output

```
I0108 08:32:28.445696 1 operator.go:209] ClusterInfo passed
I0108 08:32:28.451029 1 datastore.go:57] CheckStorageClasses checked 1 storage
classes, 0 problems found
I0108 08:32:28.451047 1 operator.go:209] CheckStorageClasses passed
I0108 08:32:28.452160 1 operator.go:209] CheckDefaultDatastore passed
I0108 08:32:28.480648 1 operator.go:271] CheckNodeDiskUUID:<host_name> passed
I0108 08:32:28.480685 1 operator.go:271] CheckNodeProviderID:<host_name> passed
```

- To view the Operator logs with the OpenShift Container Platform web console, perform the following steps:
 - a. Navigate to **Workloads** → **Pods**.
 - b. Select **openshift-cluster-storage-operator** from the **Projects** menu.
 - c. Click the link for the **vsphere-problem-detector-operator** pod.
 - d. Click the **Logs** tab on the **Pod details** page to view the logs.

23.11.5. Configuration checks run by the vSphere Problem Detector Operator

The following tables identify the configuration checks that the vSphere Problem Detector Operator runs. Some checks verify the configuration of the cluster. Other checks verify the configuration of each node in the cluster.

Table 23.98. Cluster configuration checks

Name	Description
------	-------------

Name	Description
CheckDefaultDatastore	<p>Verifies that the default datastore name in the vSphere configuration is short enough for use with dynamic provisioning.</p> <p>If this check fails, you can expect the following:</p> <ul style="list-style-type: none"> ● systemd logs errors to the journal such as Failed to set up mount unit: Invalid argument. ● systemd does not unmount volumes if the virtual machine is shut down or rebooted without draining all the pods from the node. <p>If this check fails, reconfigure vSphere with a shorter name for the default datastore.</p>
CheckFolderPermissions	<p>Verifies the permission to list volumes in the default datastore. This permission is required to create volumes. The Operator verifies the permission by listing the / and /kubevols directories. The root directory must exist. It is acceptable if the /kubevols directory does not exist when the check runs. The /kubevols directory is created when the datastore is used with dynamic provisioning if the directory does not already exist.</p> <p>If this check fails, review the required permissions for the vCenter account that was specified during the OpenShift Container Platform installation.</p>
CheckStorageClasses	<p>Verifies the following:</p> <ul style="list-style-type: none"> ● The fully qualified path to each persistent volume that is provisioned by this storage class is less than 255 characters. ● If a storage class uses a storage policy, the storage class must use one policy only and that policy must be defined.
CheckTaskPermissions	Verifies the permission to list recent tasks and datastores.
ClusterInfo	Collects the cluster version and UUID from vSphere vCenter.

Table 23.99. Node configuration checks

Name	Description
CheckNodeDiskUUID	<p>Verifies that all the vSphere virtual machines are configured with disk.enableUUID=TRUE.</p> <p>If this check fails, see the How to check 'disk.EnableUUID' parameter from VM in vSphere Red Hat Knowledgebase solution.</p>

Name	Description
CheckNodeProviderID	<p>Verifies that all nodes are configured with the ProviderID from vSphere vCenter. This check fails when the output from the following command does not include a provider ID for each node.</p> <pre>\$ oc get nodes -o custom-columns=NAME:.metadata.name,PROVIDER_ID:.spec.providerID,UUID:.status.nodeInfo.systemUUID</pre> <p>If this check fails, refer to the vSphere product documentation for information about setting the provider ID for each node in the cluster.</p>
CollectNodeESXiVersion	Reports the version of the ESXi hosts that run nodes.
CollectNodeHWVersion	Reports the virtual machine hardware version for a node.

23.11.6. About the storage class configuration check

The names for persistent volumes that use vSphere storage are related to the datastore name and cluster ID.

When a persistent volume is created, **systemd** creates a mount unit for the persistent volume. The **systemd** process has a 255 character limit for the length of the fully qualified path to the VDMK file that is used for the persistent volume.

The fully qualified path is based on the naming conventions for **systemd** and vSphere. The naming conventions use the following pattern:

```
/var/lib/kubelet/plugins/kubernetes.io/vsphere-volume/mounts/[<datastore>] 00000000-0000-0000-0000-0000000000000000/<cluster_id>-dynamic-pvc-00000000-0000-0000-0000-000000000000.vmdk
```

- The naming conventions require 205 characters of the 255 character limit.
- The datastore name and the cluster ID are determined from the deployment.
- The datastore name and cluster ID are substituted into the preceding pattern. Then the path is processed with the **systemd-escape** command to escape special characters. For example, a hyphen character uses four characters after it is escaped. The escaped value is **\x2d**.
- After processing with **systemd-escape** to ensure that **systemd** can access the fully qualified path to the VDMK file, the length of the path must be less than 255 characters.

23.11.7. Metrics for the vSphere Problem Detector Operator

The vSphere Problem Detector Operator exposes the following metrics for use by the OpenShift Container Platform monitoring stack.

Table 23.100. Metrics exposed by the vSphere Problem Detector Operator

Name	Description
vsphere_cluster_check_total	Cumulative number of cluster-level checks that the vSphere Problem Detector Operator performed. This count includes both successes and failures.
vsphere_cluster_check_errors	Number of failed cluster-level checks that the vSphere Problem Detector Operator performed. For example, a value of 1 indicates that one cluster-level check failed.
vsphere_esxi_version_total	Number of ESXi hosts with a specific version. Be aware that if a host runs more than one node, the host is counted only once.
vsphere_node_check_total	Cumulative number of node-level checks that the vSphere Problem Detector Operator performed. This count includes both successes and failures.
vsphere_node_check_errors	Number of failed node-level checks that the vSphere Problem Detector Operator performed. For example, a value of 1 indicates that one node-level check failed.
vsphere_node_hw_version_total	Number of vSphere nodes with a specific hardware version.
vsphere_vcenter_info	Information about the vSphere vCenter Server.

23.11.8. Additional resources

- [Monitoring overview](#)

CHAPTER 24. INSTALLING ON VMC

24.1. PREPARING TO INSTALL ON VMC

24.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall and plan to use Telemetry, you [configured the firewall to allow the sites](#) required by your cluster.

24.1.2. Choosing a method to install OpenShift Container Platform on VMC

You can install OpenShift Container Platform on VMC by using installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provide. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See the [Installation process](#) for more information about installer-provisioned and user-provisioned installation processes.



IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the VMC platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

24.1.2.1. Installer-provisioned infrastructure installation of OpenShift Container Platform on VMC

Installer-provisioned infrastructure allows the installation program to pre-configure and automate the provisioning of resources required by OpenShift Container Platform.

- [Installing a cluster on VMC](#) You can install OpenShift Container Platform on VMC by using installer-provisioned infrastructure installation with no customization.
- [Installing a cluster on VMC with customizations](#) You can install OpenShift Container Platform on VMC by using installer-provisioned infrastructure installation with the default customization options.
- [Installing a cluster on VMC with network customizations](#) You can install OpenShift Container Platform on installer-provisioned VMC infrastructure, with network customizations. You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.
- [Installing a cluster on VMC in a restricted network](#) You can install a cluster on VMC

infrastructure in a restricted network by creating an internal mirror of the installation release content. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.

24.1.2.2. User-provisioned infrastructure installation of OpenShift Container Platform on VMC

User-provisioned infrastructure requires the user to provision all resources required by OpenShift Container Platform.

- **Installing a cluster on VMC with user-provisioned infrastructure** You can install OpenShift Container Platform on VMC infrastructure that you provision.
- **Installing a cluster on VMC with user-provisioned infrastructure and network customizations:** You can install OpenShift Container Platform on VMC infrastructure that you provision with customized network configuration options.
- **Installing a cluster on VMC in a restricted network with user-provisioned infrastructure** OpenShift Container Platform can be installed on VMC infrastructure that you provision in a restricted network.

24.1.3. VMware vSphere infrastructure requirements

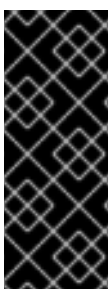
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.1. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

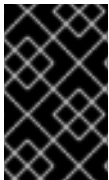


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.2. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.1.4. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

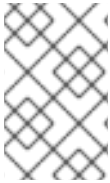
- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .

24.1.5. Uninstalling an installer-provisioned infrastructure installation of OpenShift Container Platform on VMC

- **Uninstalling a cluster on VMC that uses installer-provisioned infrastructure** You can remove a cluster that you deployed on VMC infrastructure that used installer-provisioned infrastructure.

24.2. INSTALLING A CLUSTER ON VMC

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere by deploying it to [VMware Cloud \(VMC\) on AWS](#).

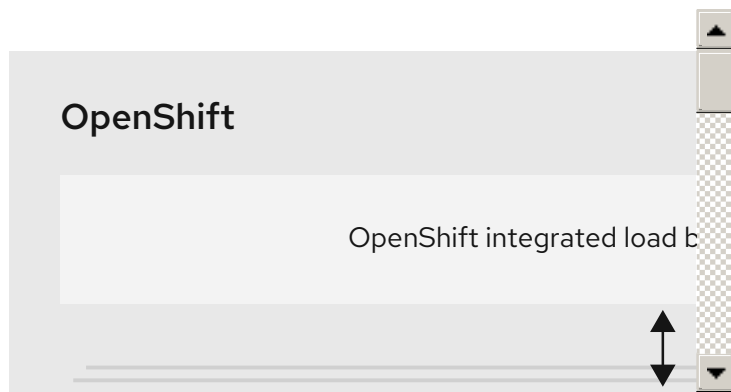


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.2.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
 - A DNS record for **api.<cluster_name>.<base_domain>** pointing to the allocated IP address.
 - A DNS record for ***.apps.<cluster_name>.<base_domain>** pointing to the allocated IP address.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the internet. This is used by nodes and applications to download container images.
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.

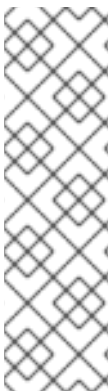
- An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**



NOTE

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool



NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.2.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.2.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

24.2.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.2.4. VMware vSphere infrastructure requirements

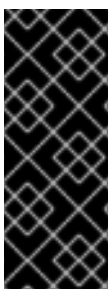
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.3. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



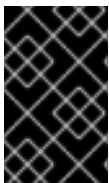
IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.4. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
-----------	----------------------------	-------------

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.2.5. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 24.5. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn

Protocol	Port	Description
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.6. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.7. Ports used for control plane machine to control plane machine communications

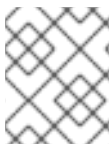
Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

24.2.6. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.2.7. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 24.1. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.A ttachTag InventoryService.Tagging.C reateCategory InventoryService.Tagging.C reateTag InventoryService.Tagging.D deleteCategory InventoryService.Tagging.D deleteTag InventoryService.Tagging.E ditCategory InventoryService.Tagging.E ditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk

vSphere object for role	When required	VirtualMachine.Config.Rena Required privileges in vSphere API VirtualMachine.Config.Rese tGuestInfo VirtualMachine.Config.Reso urce VirtualMachine.Config.Setti ngs VirtualMachine.Config.Upgr adeVirtualHardware VirtualMachine.Interact.Gue stControl VirtualMachine.Interact.Pow erOff VirtualMachine.Interact.Pow erOn VirtualMachine.Interact.Res et VirtualMachine.Inventory.Cr eate VirtualMachine.Inventory.Cr eateFromExisting VirtualMachine.Inventory.D elete VirtualMachine.Provisionin g.Clone VirtualMachine.Provisionin g.MarkAsTemplate VirtualMachine.Provisionin g.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	InventoryService.Tagging.O bjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.2. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change Virtual machine".Change Configuration".Reset guest GUI Configuration".Reset guest information"
		"Virtual machine".Change Configuration".Change resource" "Virtual machine".Change Configuration".Change Settings" "Virtual machine".Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine".Edit Inventory".Create new" "Virtual machine".Edit Inventory".Create from existing" "Virtual machine".Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource.Assign virtual machine to resource pool" VApp.Import "Virtual machine".Change Configuration".Add existing disk" "Virtual machine".Change Configuration".Add new disk" "Virtual machine".Change

vSphere object for role	When required	Configuration". "Add or Remove device" Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from Existing Virtual machine". "Edit Virtual machine". "Remove Virtual machine". Provisioning. "Clone virtual machine" Virtual machine". Provisioning. "Deploy template" Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.3. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 24.8. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

24.2.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

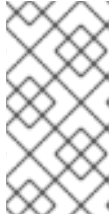
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

- If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

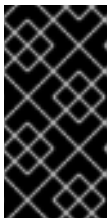
- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

24.2.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

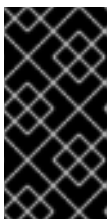


IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

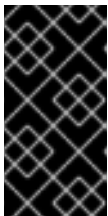
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

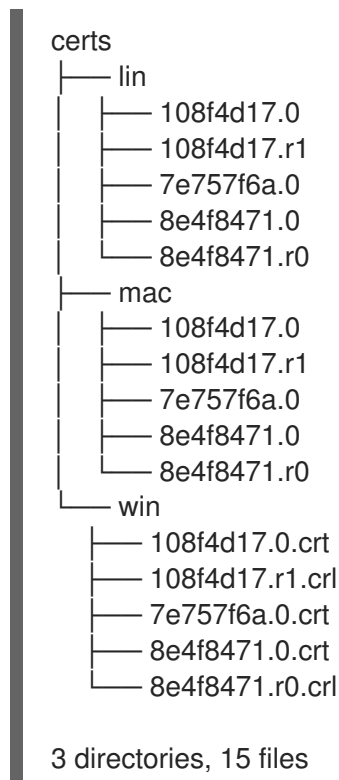
5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

24.2.10. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

24.2.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

When you have configured your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host that is co-located in the VMC environment. The installation program and control plane automates the process

of deploying and managing the resources needed for the OpenShift Container Platform cluster.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1  
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
 - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Provide values at the prompts:
 - a. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- b. Select **vsphere** as the platform to target.

- c. Specify the name of your vCenter instance.
- d. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.

**IMPORTANT**

Some VMware vCenter Single Sign-On (SSO) environments with Active Directory (AD) integration might primarily require you to use the traditional login method, which requires the **<domain>** construct.

To ensure that vCenter account permission checks complete properly, consider using the User Principal Name (UPN) login method, such as **<username>@<fully_qualified_domainname>**.

- e. Select the data center in your vCenter instance to connect to.
- f. Select the default vCenter datastore to use.

**NOTE**

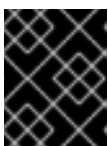
Datastore and cluster names cannot exceed 60 characters; therefore, ensure the combined string length does not exceed the 60 character limit.

- g. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
- h. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
- i. Enter the virtual IP address that you configured for control plane API access.
- j. Enter the virtual IP address that you configured for cluster ingress.
- k. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
- l. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

**NOTE**

Datastore and cluster names cannot exceed 60 characters; therefore, ensure the combined string length does not exceed the 60 character limit.

- m. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .

**IMPORTANT**

Use the **openshift-install** command from the bastion hosted in the VMC environment.

+

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.

**IMPORTANT**

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

24.2.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

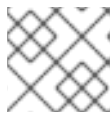
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

24.2.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

-

- 1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.2.14. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

24.2.14.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from **Removed** to **Managed**. When this has completed, you must configure storage.

24.2.14.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.2.14.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

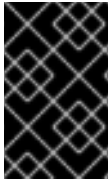
```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

24.2.14.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
```

```
- ReadWriteOnce 3
resources:
  requests:
    storage: 100Gi 4
```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

24.2.15. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.

4. Create a backup of the cloned volume.
5. Delete the cloned volume.

24.2.16. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

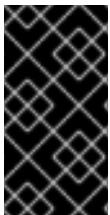
After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.2.17. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

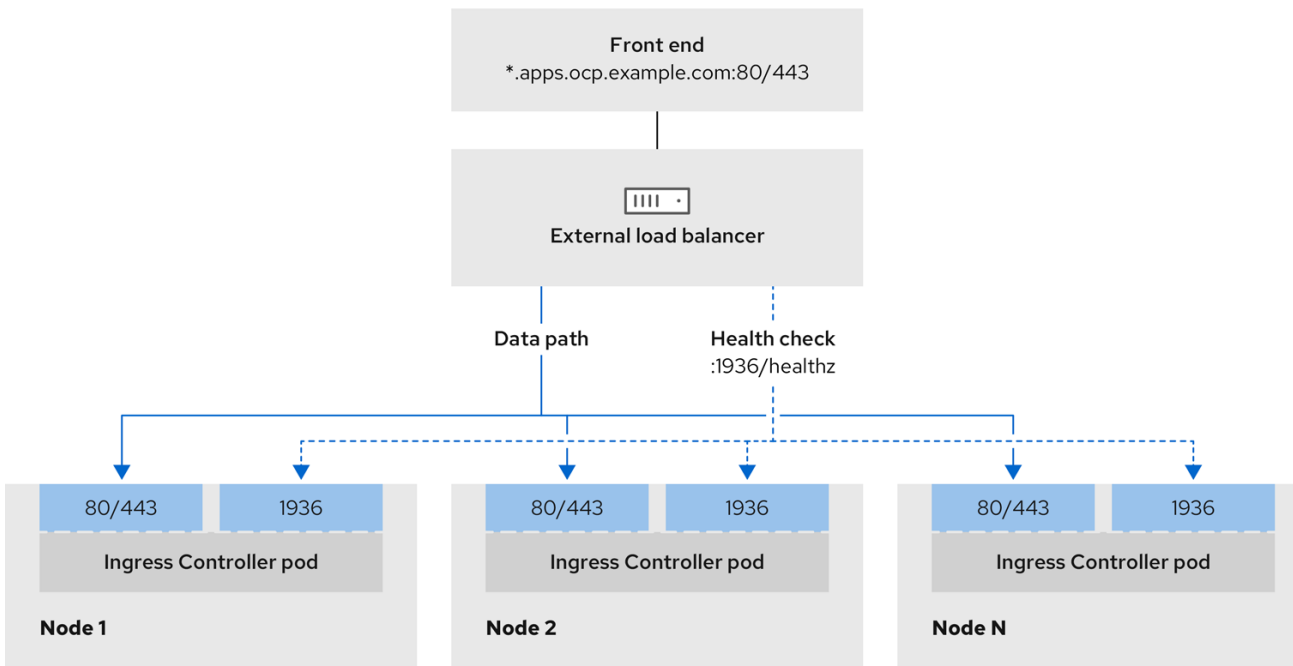
The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

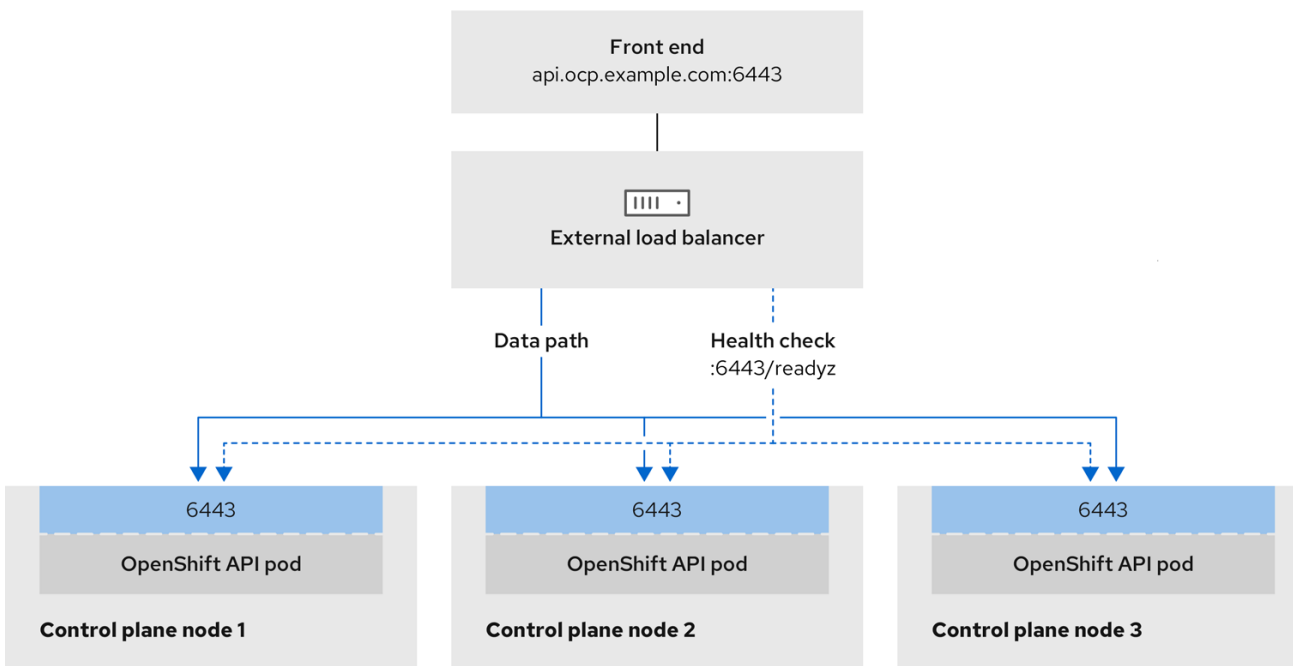
You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 24.1. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



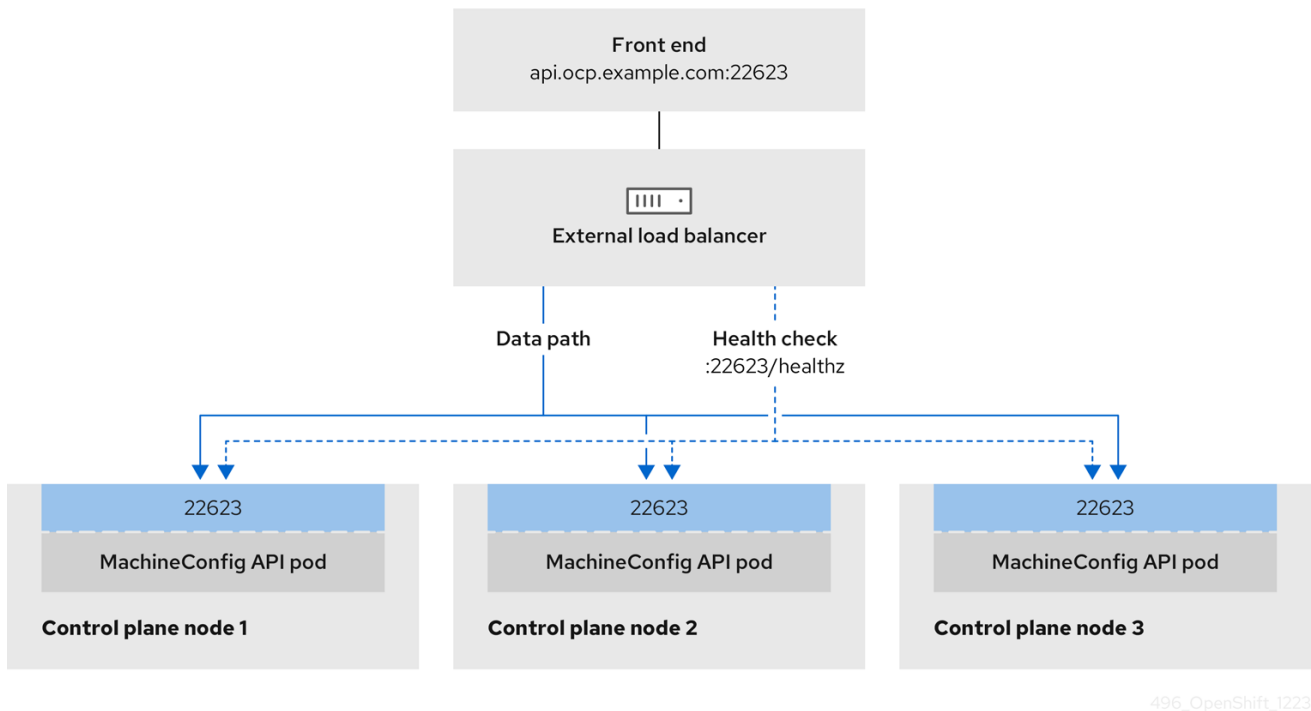
496_OpenShift_1223

Figure 24.2. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment



496_OpenShift_1223

Figure 24.3. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a /27 or /28, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

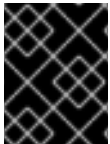
Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

24.2.17.1. Configuring an external load balancer

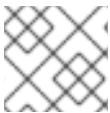
You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",

```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie: 1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/; HttpOnly; Secure; SameSite=None
cache-control: private

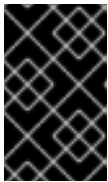
```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

24.2.18. Next steps

- [Customize your cluster.](#)

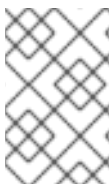
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.3. INSTALLING A CLUSTER ON VMC WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance using installer-provisioned infrastructure by deploying it to [VMware Cloud \(VMC\) on AWS](#) .

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

To customize the OpenShift Container Platform installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

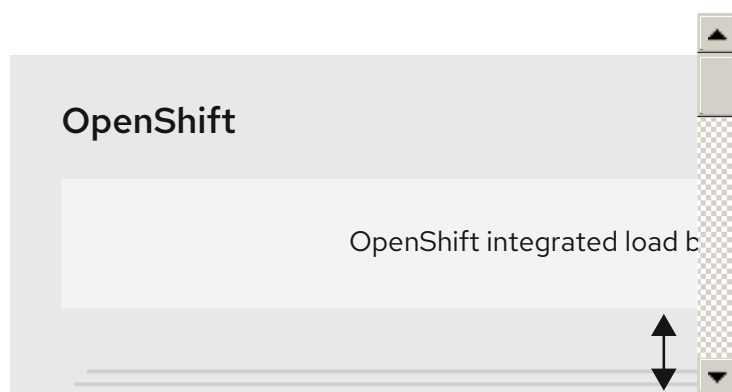


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.3.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
 - A DNS record for **api.<cluster_name>.<base_domain>** pointing to the allocated IP address.

- A DNS record for ***.apps.<cluster_name>.<base_domain>** pointing to the allocated IP address.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the internet. This is used by nodes and applications to download container images.
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**

**NOTE**

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.3.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

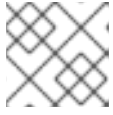
To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.3.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

24.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.3.4. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.9. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

**IMPORTANT**

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.10. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.3.5. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 24.11. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests
TCP	1936	Metrics

Protocol	Port	Description
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.12. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.13. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

24.3.6. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.3.7. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 24.4. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API VirtualMachine.Config.AdvancedConfig
		VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk

vSphere object for role	When required delete privileges are optional if your cluster does not use the Machine API.	Required privileges in vSphere API
		VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.5. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine".Change Configuration".Change resource" "Virtual machine".Change Configuration".Change Settings" "Virtual machine".Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine".Edit Inventory".Create new" "Virtual machine".Edit Inventory".Create from existing" "Virtual machine".Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine".Change Configuration".Add existing disk" "Virtual machine".Change Configuration".Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
		<p>"Virtual machine"."Change Configuration". "Add or remove device"</p> <p>"Virtual machine"."Change Configuration". "Advanced configuration"</p> <p>"Virtual machine"."Change Configuration". "Set annotation"</p> <p>"Virtual machine"."Change Configuration". "Change CPU count"</p> <p>"Virtual machine"."Change Configuration". "Extend virtual disk"</p> <p>"Virtual machine"."Change Configuration". "Acquire disk lease"</p> <p>"Virtual machine"."Change Configuration". "Modify device settings"</p> <p>"Virtual machine"."Change Configuration". "Change Memory"</p> <p>"Virtual machine"."Change Configuration". "Remove disk"</p> <p>"Virtual machine"."Change Configuration". "Rename"</p> <p>"Virtual machine"."Change Configuration". "Reset guest information"</p> <p>"Virtual machine"."Change Configuration". "Change resource"</p> <p>"Virtual machine"."Change Configuration". "Change Settings"</p> <p>"Virtual machine"."Change Configuration". "Upgrade virtual machine compatibility"</p> <p>"Virtual machine".Interaction."Guest operating system management by VIX API"</p> <p>"Virtual machine".Interaction."Power off"</p> <p>"Virtual machine".Interaction."Power on"</p> <p>"Virtual machine".Interaction."Reset"</p> <p>"Virtual machine"."Edit Inventory". "Create new"</p>

vSphere object for role	When required	"Virtual machine". "Edit Inventory". "Create from existing"
		"Virtual machine". "Edit Inventory". "Remove" "Virtual machine". Provisioning. "Clone virtual machine" "Virtual machine". Provisioning. "Deploy template" "Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.6. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 24.14. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

24.3.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

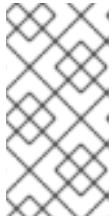
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

- If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

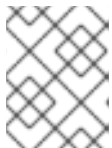
- View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

- Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

- Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

24.3.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

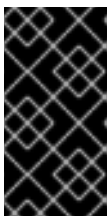


IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

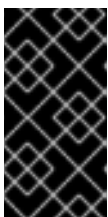
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

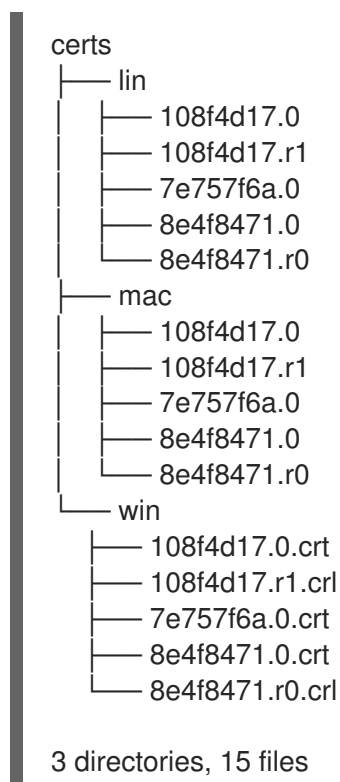
5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

24.3.10. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

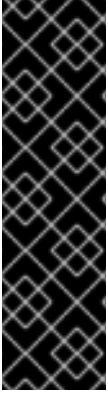
```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

24.3.11. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.15. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.3.12. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

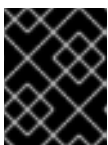
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
 - iii. Specify the name of your vCenter instance.
 - iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
 - v. Select the data center in your vCenter instance to connect to.
 - vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

24.3.12.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

24.3.12.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 24.16. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or <code>{}</code> . For additional information about platform . <code><platform></code> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

24.3.12.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 24.17. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>


24.3.12.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 24.18. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

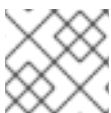
Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">  </div> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <div data-bbox="485 936 595 1288" style="border: 1px solid black; padding: 5px;">  </div> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div style="display: flex; align-items: flex-start;"> <div style="width: 60px; height: 150px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="width: 60px; height: 60px; background: repeating-linear-gradient(45deg, transparent, transparent 2px, black 2px, black 4px); margin-right: 10px;"></div> <div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

24.3.12.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table.





NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.19. Additional VMware vSphere cluster parameters

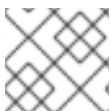
Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String

Parameter	Description	Values
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String
apiVIPs	The virtual IP (VIP) address that you configured for control plane API access.  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .

Parameter	Description	Values
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

24.3.12.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

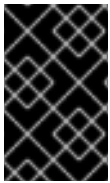
Table 24.20. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware-<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer

Parameter	Description	Values
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

24.3.12.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.21. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String

Parameter	Description	Values
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String

24.3.12.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 3
  platform:
    vsphere: 3
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: 4
  name: master
  replicas: 3
  platform:
    vsphere: 5
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster 6
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    resourcePool: resource_pool 7
    diskType: thin 8
    network: VM_Network
    cluster: vsphere_cluster_name 9
  apiVIPs:
    - api_vip
  ingressVIPs:
    - ingress_vip
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'

```

1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

- 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
- 6 The cluster name that you specified in your DNS records.
- 7 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
- 8 The vSphere disk provisioning method.
- 9 The vSphere cluster to install the OpenShift Container Platform cluster in.



NOTE

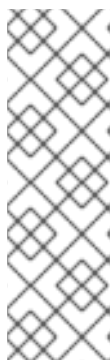
In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

24.3.12.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

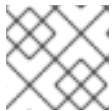
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```

```

additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



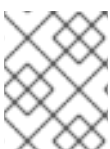
NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

24.3.12.4. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).



IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

- Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

■


```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

- To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

- To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

- Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample `install-config.yaml` file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: 2
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: 3
    - "<machine_pool_zone_1>"
    - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> 4
    username: <username> 5
    password: <password> 6
```

```

datacenter: datacenter 7
defaultDatastore: datastore 8
folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
cluster: cluster 10
resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
diskType: thin
failureDomains: 12
- name: <machine_pool_zone_1> 13
  region: <region_tag_1> 14
  zone: <zone_tag_1> 15
  topology: 16
    datacenter: <datacenter1> 17
    computeCluster: "/<datacenter1>/host/<cluster1>" 18
    resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
    networks: 20
    - <VM_Network1_name>
    datastore: "/<datacenter1>/datastore/<datastore1>" 21
- name: <machine_pool_zone_2>
  region: <region_tag_2>
  zone: <zone_tag_2>
  topology:
    datacenter: <datacenter2>
    computeCluster: "/<datacenter2>/host/<cluster2>"
    networks:
    - <VM_Network2_name>
    datastore: "/<datacenter2>/datastore/<datastore2>"
    resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
    folder: "/<datacenter2>/vm/<folder2>"
# ...

```

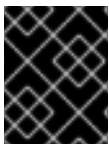
- 1** You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2** **3** An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4** **5** **6** **7** **8** **9** **10** **11** The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13** Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14** You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15** You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.

- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.3.13. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

When you have configured your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host that is co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



IMPORTANT

Use the **openshift-install** command from the bastion hosted in the VMC environment.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/./openshift_install.log**.

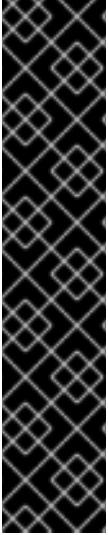


IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

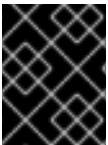


IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

24.3.14. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

24.3.15. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.3.16. Creating registry storage

After you install the cluster, you must create storage for the Registry Operator.

24.3.16.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

24.3.16.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.3.16.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```


Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

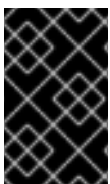
```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	False

24.3.16.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

■

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage ❶
  namespace: openshift-image-registry ❷
spec:
  accessModes:
  - ReadWriteOnce ❸
  resources:
    requests:
      storage: 100Gi ❹
```

- ❶ A unique name that represents the **PersistentVolumeClaim** object.
- ❷ The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- ❸ The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- ❹ The size of the persistent volume claim.

- b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: ❶
```

- ❶ By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

24.3.17. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

24.3.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

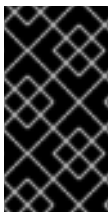
After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.3.19. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

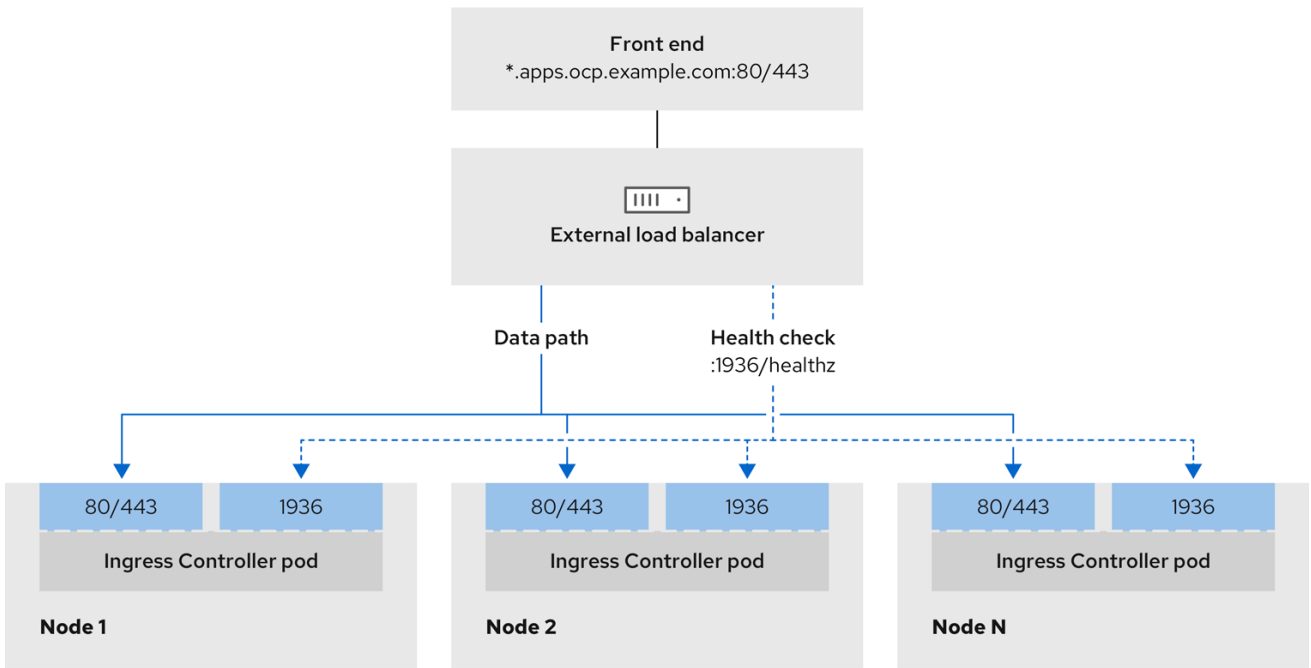
The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

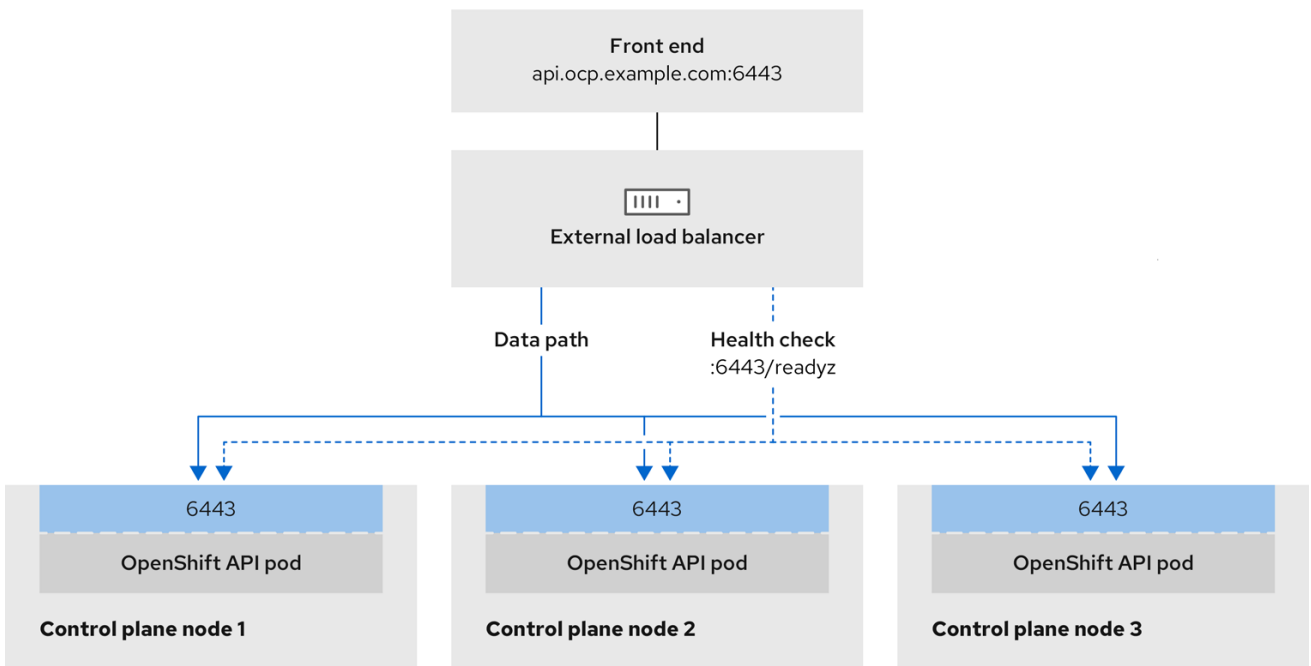
You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 24.4. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



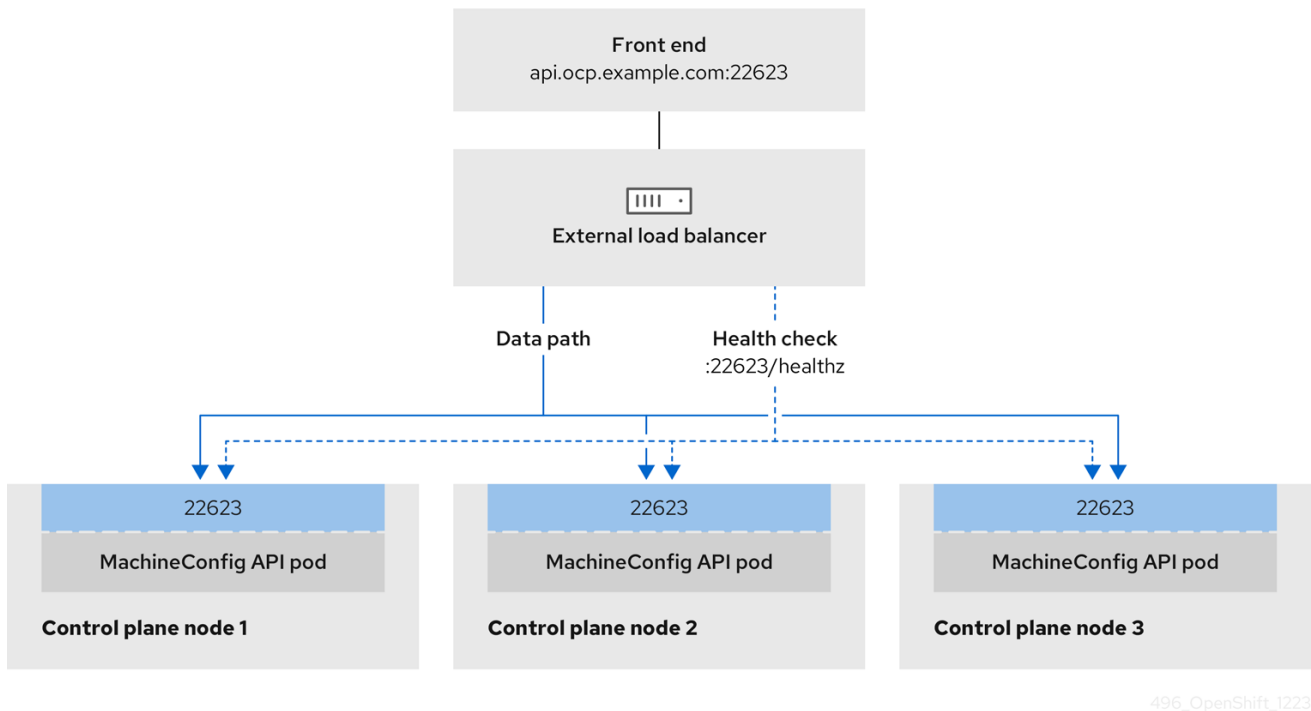
496_OpenShift_1223

Figure 24.5. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment



496_OpenShift_1223

Figure 24.6. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

24.3.19.1. Configuring an external load balancer

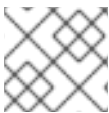
You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
}
```



```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-token=UIYWOyQ62LWjw2h003xtYSKIh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie: 1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/; HttpOnly; Secure; SameSite=None
cache-control: private

```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

24.3.20. Next steps

- [Customize your cluster.](#)

- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.4. INSTALLING A CLUSTER ON VMC WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance using installer-provisioned infrastructure with customized network configuration options by deploying it to [VMware Cloud \(VMC\) on AWS](#).

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

By customizing your OpenShift Container Platform network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing VXLAN configurations. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster. You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

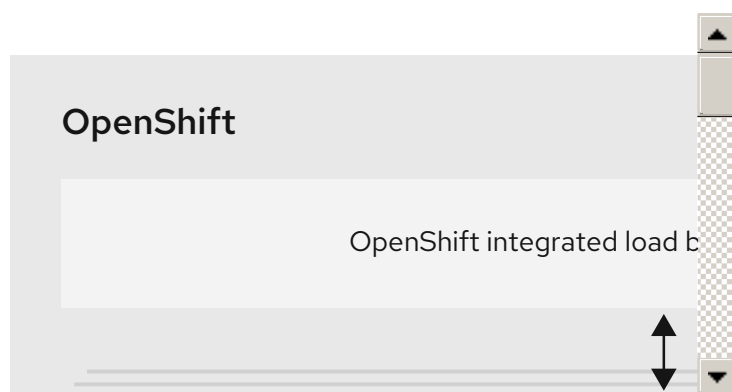


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.4.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
 - A DNS record for **api.<cluster_name>.<base_domain>** pointing to the allocated IP address.
 - A DNS record for ***.apps.<cluster_name>.<base_domain>** pointing to the allocated IP address.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the internet. This is used by nodes and applications to download container images.
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**



NOTE

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.

- The **openshift-install** installation program
- The OpenShift CLI (**oc**) tool



NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.4.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

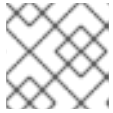
- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.4.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).

- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

24.4.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.4.4. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.22. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

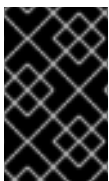


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.23. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.4.5. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 24.24. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests

Protocol	Port	Description
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.25. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.26. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

24.4.6. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later

- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.4.7. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 24.7. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API VirtualMachine.Config.AdvancedConfig
		VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.Add

vSphere object for role	When required delete privileges are optional if your cluster does not use the Machine API.	NewDisk Required privileges in vSphere API VirtualMachine.Config.AddRemoveDevice
		VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.8. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration".Change resource" "Virtual machine"."Change Configuration".Change Settings" "Virtual machine"."Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration".Add existing disk" "Virtual machine"."Change Configuration".Add new disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Add or Remove device" GUI "Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from Existing Virtual machine". "Edit Virtual machine". "Remove Virtual machine". "Provisioning. "Clone virtual machine" "Virtual machine". "Provisioning. "Deploy template" "Virtual machine". "Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.9. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 24.27. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

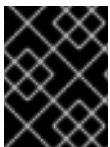
Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

24.4.8. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

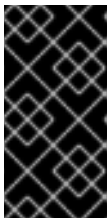
- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

24.4.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space.

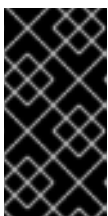


IMPORTANT

If you attempt to run the installation program on macOS, a known issue related to the **golang** compiler causes the installation of the OpenShift Container Platform cluster to fail. For more information about this issue, see the section named "Known Issues" in the *OpenShift Container Platform 4.12 release notes* document.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

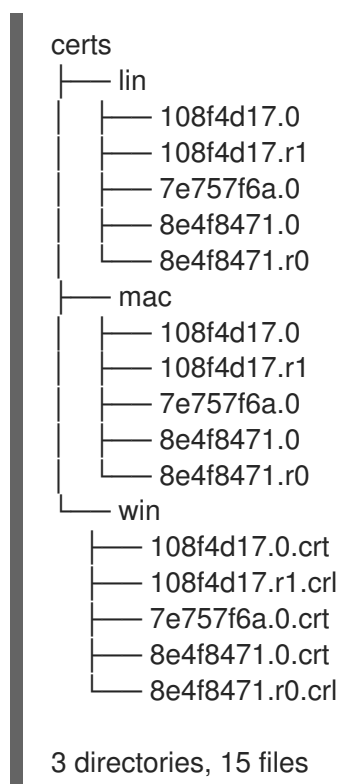
5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

24.4.10. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

24.4.11. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

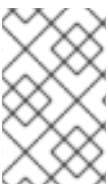
For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.28. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

Datacenter (region)	Cluster (zone)	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.4.12. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```

$ ./openshift-install create install-config --dir <installation_directory> 1

```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:

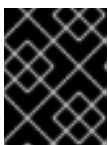
- i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
 - iii. Specify the name of your vCenter instance.
 - iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
 - v. Select the data center in your vCenter instance to connect to.
 - vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.
 3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

24.4.12.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the **install-config.yaml** file.

24.4.12.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 24.29. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .

Parameter	Description	Values
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud , aws , baremetal , azure , gcp , ibmcloud , nutanix , openstack , ovirt , vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

24.4.12.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.





NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 24.30. Network parameters

Parameter	Description	Values
-----------	-------------	--------

Parameter	Description	Values
networking	The configuration for the cluster network.	Object  NOTE You cannot modify parameters specified by the networking object after installation.
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  <p>NOTE</p> <p>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</p>

24.4.12.1.3. Optional configuration parameters



Optional installation configuration parameters are described in the following table:

Table 24.31. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String

Parameter	Description	Values
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or hyperthreading , on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .

Parameter	Description	Values
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: flex-start;"> <div style="background-color: black; width: 20px; height: 40px; margin-right: 10px;"></div> <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <p> NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> <p> NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="486 586 592 1393" style="background-color: black; color: white; padding: 5px; margin-bottom: 10px;"> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> </div> <div data-bbox="486 1440 592 1637" style="background-color: black; color: white; padding: 5px;"> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p> </div>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

24.4.12.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table.




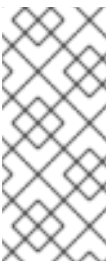
NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.32. Additional VMware vSphere cluster parameters

Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String

Parameter	Description	Values
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String
apiVIPs	The virtual IP (VIP) address that you configured for control plane API access.  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .

Parameter	Description	Values
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

24.4.12.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.33. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware-<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer

Parameter	Description	Values
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

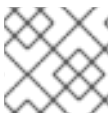
24.4.12.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.34. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String

Parameter	Description	Values
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String

24.4.12.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 3
  platform:
    vsphere: 3
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8192
      osDisk:
        diskSizeGB: 120
controlPlane: 4
  name: master
  replicas: 3
  platform:
    vsphere: 5
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster 6
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 7
  serviceNetwork:
    - 172.30.0.0/16
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: folder
    resourcePool: resource_pool 8
    diskType: thin 9
    network: VM_Network
    cluster: vsphere_cluster_name 10
  apiVIPs:
    - api_vip
  ingressVIPs:
    - ingress_vip

```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
- 6 The cluster name that you specified in your DNS records.
- 8 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
- 9 The vSphere disk provisioning method.
- 10 The vSphere cluster to install the OpenShift Container Platform cluster in.
- 7 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.



NOTE

In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

24.4.12.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

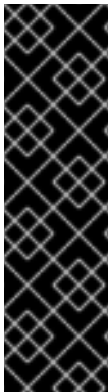
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

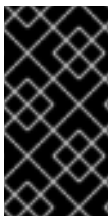
24.4.12.4. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.

**IMPORTANT**

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

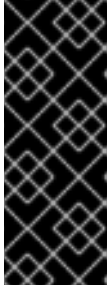
For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

**IMPORTANT**

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.

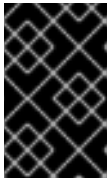


NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```

apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
    datacenter: datacenter ❼
    defaultDatastore: datastore ❽
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" ❾
    cluster: cluster ❿
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" ❶❶
    diskType: thin
    failureDomains: ❶❷
      - name: <machine_pool_zone_1> ❶❸
        region: <region_tag_1> ❶❹
        zone: <zone_tag_1> ❶❺
        topology: ❶❻
          datacenter: <datacenter1> ❶❼
          computeCluster: "/<datacenter1>/host/<cluster1>" ❶❶❶
          resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" ❶❶❷
          networks: ❶❶❸
            - <VM_Network1_name>
          datastore: "/<datacenter1>/datastore/<datastore1>" ❶❶❹
      - name: <machine_pool_zone_2>
        region: <region_tag_2>
        zone: <zone_tag_2>
        topology:
          datacenter: <datacenter2>
          computeCluster: "/<datacenter2>/host/<cluster2>"
          networks:
            - <VM_Network2_name>
          datastore: "/<datacenter2>/datastore/<datastore2>"

```

```
resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
folder: "/<datacenter2>/vm/<folder2>"
# ...
```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**. If you do not specify a value, resources are installed in the root of the cluster **/example_datacenter/host/example_cluster/Resources**.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.4.13. Network configuration phases

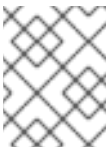
There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**

For more information on these fields, refer to *Installation configuration parameters*.



NOTE

Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.



IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by libVirt. You cannot use this range or any range that overlaps with this range for any networks in your cluster.

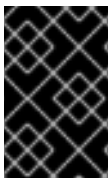
Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify advanced network configuration.

You cannot override the values specified in phase 1 in the **install-config.yaml** file during phase 2. However, you can further customize the network plugin during phase 2.

24.4.14. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

24.4.15. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

24.4.15.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 24.35. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.

Field	Type	Description
spec.kubeProxy Config	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 24.36. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 24.37. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>

Example OpenShift SDN configuration


```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 24.38. `ovnKubernetesConfig` object

Field	Type	Description
-------	------	-------------

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 24.39. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 24.40. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

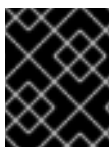
Table 24.41. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

24.4.16. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

When you have configured your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host that is co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

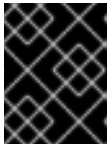
Procedure

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



IMPORTANT

Use the **openshift-install** command from the bastion hosted in the VMC environment.



NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

24.4.17. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

-

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

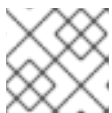
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

24.4.18. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.4.19. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

24.4.19.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

24.4.19.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.4.19.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1** Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

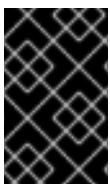
```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	False 6h50m

24.4.19.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

■

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4 The size of the persistent volume claim.

- b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

24.4.20. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

24.4.21. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

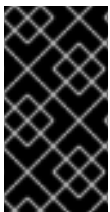
After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.4.22. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

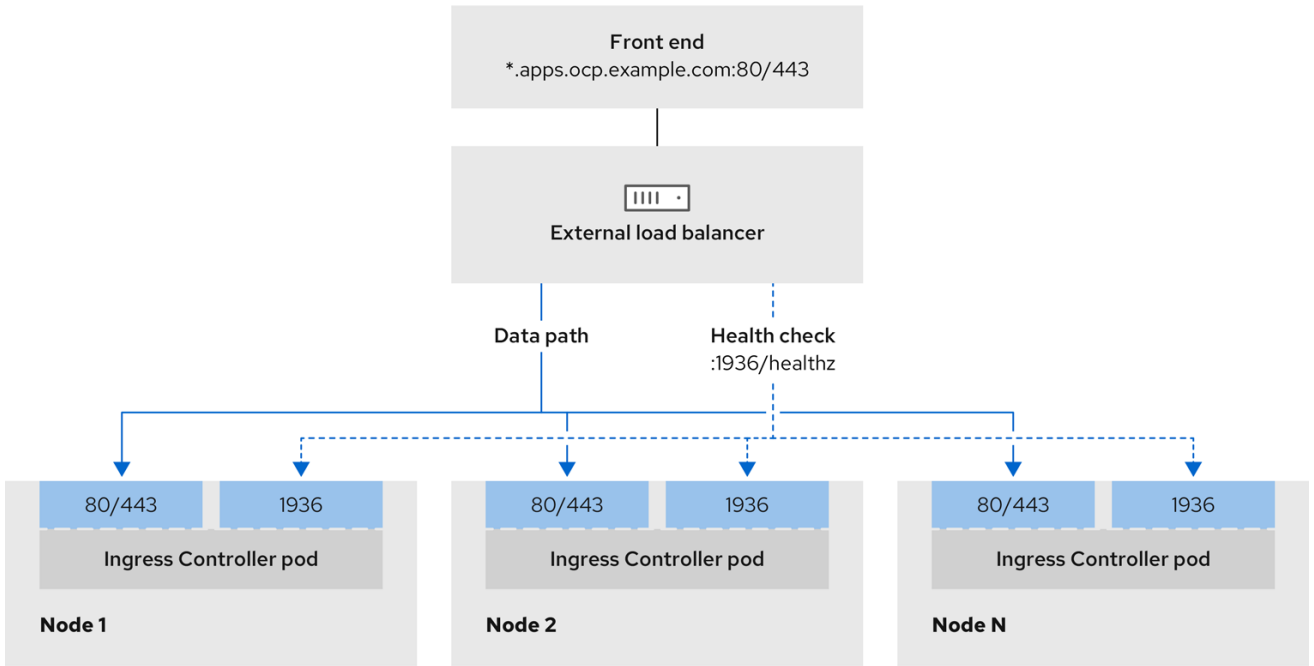
The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for an external load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

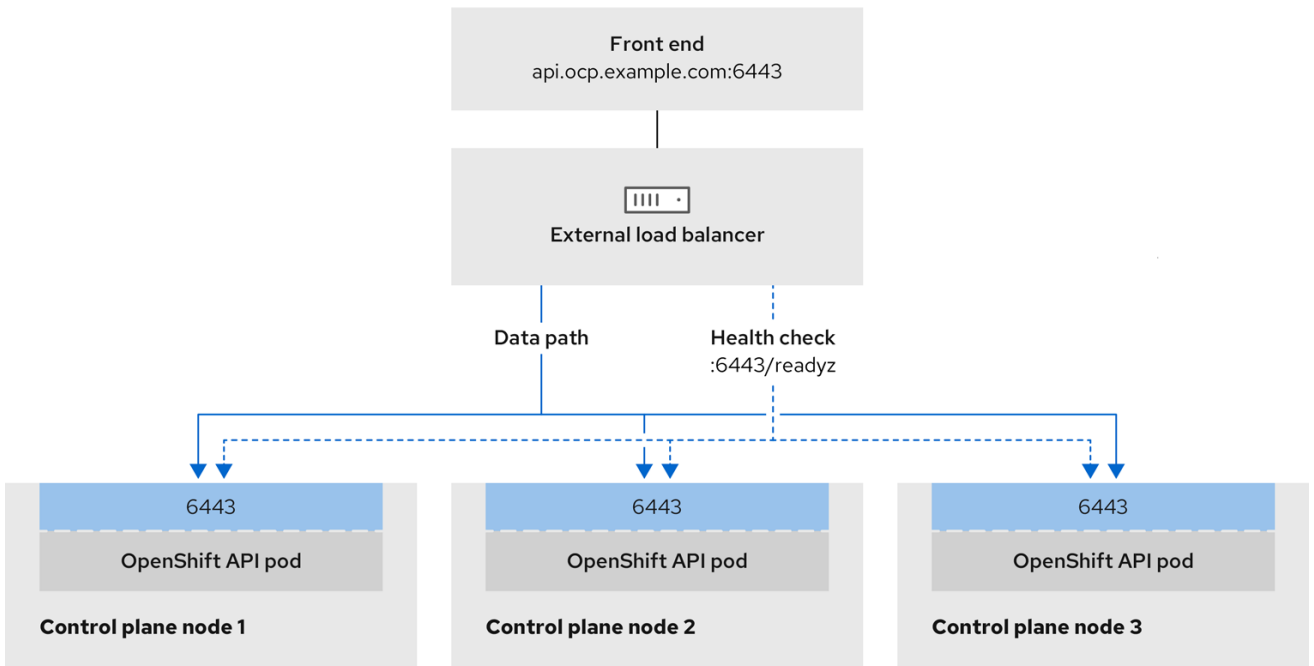
You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 24.7. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



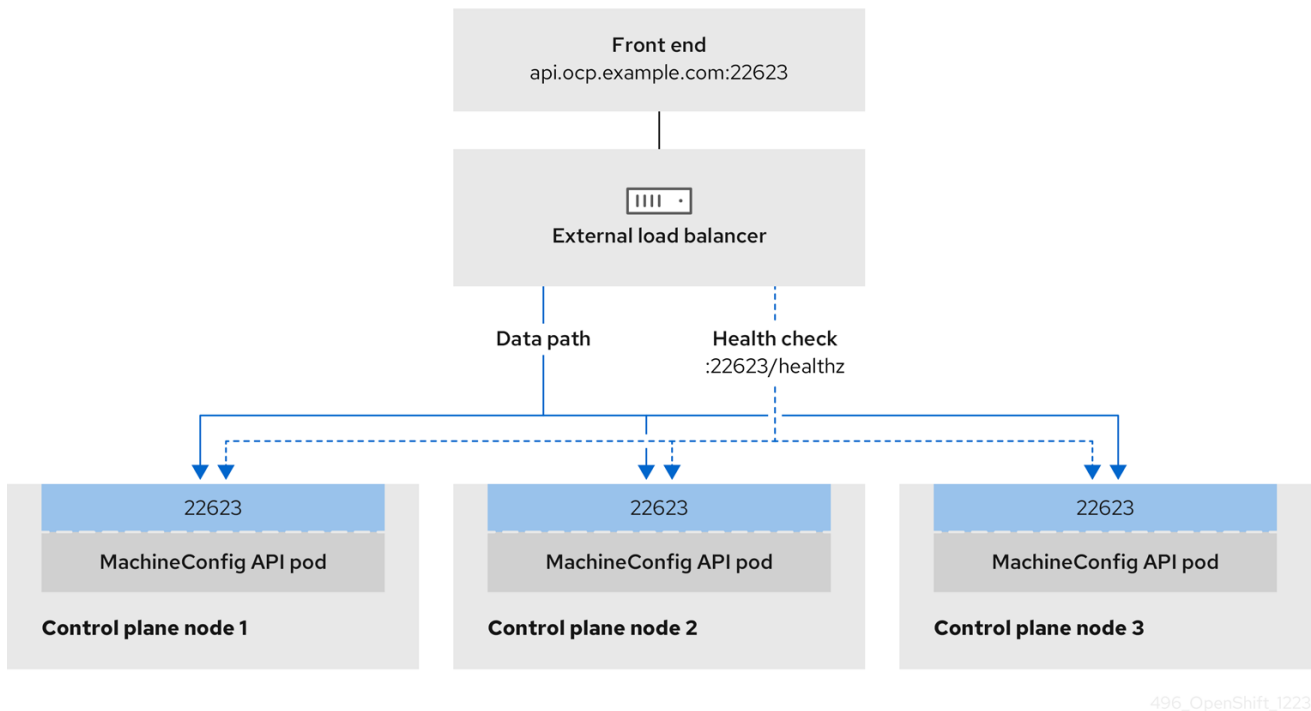
496_OpenShift_1223

Figure 24.8. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment



496_OpenShift_1223

Figure 24.9. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



496_OpenShift_1223

The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.
- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.

- Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

24.4.22.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
 - Port 6443 provides access to the OpenShift API service.
 - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.

- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```
#...
listen my-cluster-api-6443
  bind 192.168.1.100:6443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
```

```

server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
  bind 192.168.1.100:22623
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
  server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
  server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
  bind 192.168.1.100:443
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
  bind 192.168.1.100:80
  mode tcp
  balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:
 - a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
}
```

```

"gitTreeState": "clean",
"buildDate": "2019-01-09T06:44:10Z",
"goVersion": "go1.10.3",
"compiler": "gc",
"platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dGLgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie: 1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/; HttpOnly; Secure; SameSite=None
cache-control: private

```

- 3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

- 4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```


- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

24.4.23. Next steps

- [Customize your cluster.](#)

- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.5. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure in a restricted network by deploying it to [VMware Cloud \(VMC\) on AWS](#) .

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

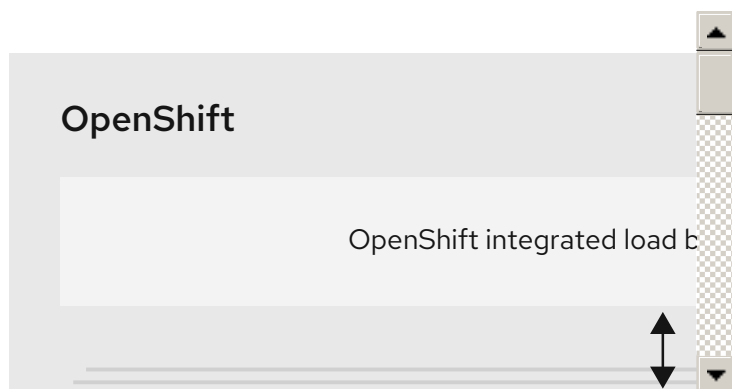


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.5.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
 - A DNS record for **api.<cluster_name>.<base_domain>** pointing to the allocated IP address.
 - A DNS record for ***.apps.<cluster_name>.<base_domain>** pointing to the allocated IP address.

- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**



NOTE

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.5.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

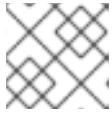
24.5.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtained the **imageContentSources** data for your version of OpenShift Container Platform.

**IMPORTANT**

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

If you are configuring a proxy, be sure to also review this site list.

24.5.3. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

24.5.3.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

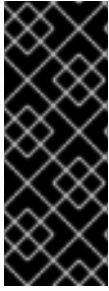
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

24.5.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.5.5. VMware vSphere infrastructure requirements

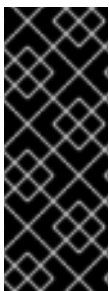
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.42. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.43. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
-----------	----------------------------	-------------

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.5.6. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate.

Review the following details about the required network ports.

Table 24.44. Ports used for all-machine to all-machine communications

Protocol	Port	Description
VRRP	N/A	Required for keepalived
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn

Protocol	Port	Description
UDP	4789	virtual extensible LAN (VXLAN)
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.45. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.46. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

24.5.7. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.5.8. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 24.10. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.Add NewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.Storage Resource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.Add NewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.O bjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.O bjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.Add ExistingDisk VirtualMachine.Config.Add NewDisk VirtualMachine.Config.Add RemoveDevice VirtualMachine.Config.Adva ncedConfig VirtualMachine.Config.Anno tation VirtualMachine.Config.CPU Count VirtualMachine.Config.Disk Extend VirtualMachine.Config.Disk Lease VirtualMachine.Config.Edit Device VirtualMachine.Config.Mem ory VirtualMachine.Config.Rem oveDisk

vSphere object for role	When required	VirtualMachine.Config.Rena Required privileges in vSphere API VirtualMachine.Config.Rese tGuestInfo
		VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.Delete privileges are optional if your cluster does not use the Machine API.	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.Rem

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.11. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change Configuration".Reset guest GUI information"
		"Virtual machine".Change Configuration".Change resource" "Virtual machine".Change Configuration".Change Settings" "Virtual machine".Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine".Edit Inventory".Create new" "Virtual machine".Edit Inventory".Create from existing" "Virtual machine".Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource.Assign virtual machine to resource pool" VApp.Import "Virtual machine".Change Configuration".Add existing disk" "Virtual machine".Change Configuration".Add new disk" "Virtual machine".Change

vSphere object for role	When required	Configuration". "Add or Remove device Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from Existing Virtual machine". "Edit Virtual machine". "Remove Virtual machine". "Provisioning. "Clone virtual machine" "Virtual machine". "Provisioning. "Deploy template" "Virtual machine". "Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.12. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. The VM in your restricted network must have access to vCenter so that it can provision and manage nodes, persistent volume claims (PVCs), and other resources. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 24.47. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>..	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Component	Record	Description
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

24.5.9. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

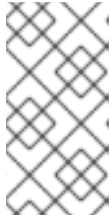
You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.

**NOTE**

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

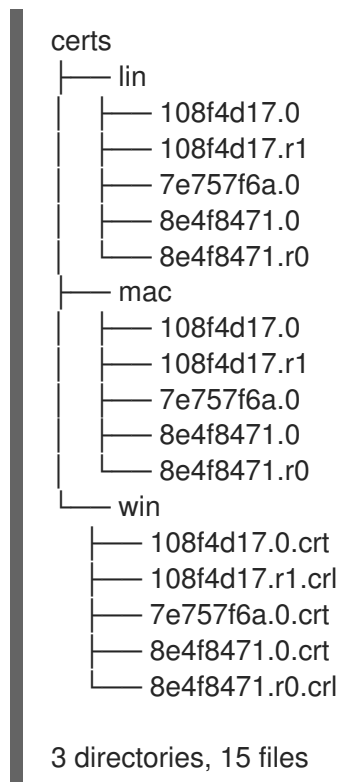
- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

24.5.10. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter's API, you must add your vCenter's trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter's root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The **<vCenter>/certs/download.zip** file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:



3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

24.5.11. Creating the RHCOS image for restricted network installations

Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network VMware vSphere environment.

Prerequisites

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

Procedure

1. Log in to the Red Hat Customer Portal's [Product Downloads page](#).
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.12 for RHEL 8.



IMPORTANT

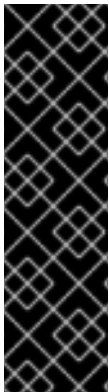
The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - vSphere** image.
4. Upload the image you downloaded to a location that is accessible from the bastion server.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

24.5.12. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- Failure domain: Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- Region: Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- Zone: Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.48. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.5.13. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the **imageContentSources** values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.
- Obtain service principal permissions at the subscription level.

Procedure

1. Create the **install-config.yaml** file.
 - a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

- b. At the prompts, provide the configuration details for your cloud:
 - i. Optional: Select an SSH key to use to access your cluster machines.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- ii. Select **vsphere** as the platform to target.
- iii. Specify the name of your vCenter instance.
- iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
The installation program connects to your vCenter instance.
- v. Select the data center in your vCenter instance to connect to.

- vi. Select the default vCenter datastore to use.
 - vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
 - viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.
 - ix. Enter the virtual IP address that you configured for control plane API access.
 - x. Enter the virtual IP address that you configured for cluster ingress.
 - xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.
 - xii. Enter a descriptive name for your cluster. The cluster name you enter must match the cluster name you specified when configuring the DNS records.
 - xiii. Paste the [pull secret from the Red Hat OpenShift Cluster Manager](#) .
2. In the **install-config.yaml** file, set the value of **platform.vsphere.clusterOSImage** to the image location or name. For example:

```
platform:
  vsphere:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-vmware.x86_64.ova?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.
 - a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email": "you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

- b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  /-----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

- c. Add the image content resources, which resemble the following YAML excerpt:

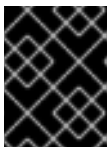
```

imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: registry.redhat.io/ocp/release

```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

4. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

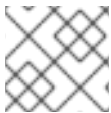


IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

24.5.13.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the **install-config.yaml** installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the **install-config.yaml** file to provide more details about the platform.



NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

24.5.13.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 24.49. Required parameters

Parameter	Description	Values
apiVersion	The API version for the install-config.yaml content. The current version is v1 . The installation program may also support older API versions.	String

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name> . <baseDomain> format.	A fully-qualified domain or subdomain name, such as example.com .
metadata	Kubernetes resource ObjectMeta , from which only the name parameter is consumed.	Object
metadata.name	The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}} . {{.baseDomain}} .	String of lowercase letters and hyphens (-), such as dev .
platform	The configuration for the specific platform upon which to perform the installation: alibabacloud, aws, baremetal, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {} . For additional information about platform . <platform> parameters, consult the table for your specific platform that follows.	Object

Parameter	Description	Values
pullSecret	Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

24.5.13.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.




NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 24.50. Network parameters

Parameter	Description	Values
networking	The configuration for the cluster network.	Object <div style="display: flex; align-items: center; margin-top: 10px;"> <div> <p>NOTE</p> <p>You cannot modify parameters specified by the networking object after installation.</p> </div> </div>

Parameter	Description	Values
networking.networkType	The Red Hat OpenShift Networking network plugin to install.	Either OpenShiftSDN or OVNKubernetes . OpenShiftSDN is a CNI plugin for all-Linux networks. OVNKubernetes is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is OVNKubernetes .
networking.clusterNetwork	The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23 . If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	Required if you use networking.clusterNetwork . An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32 .
networking.clusterNetwork.hostPrefix	The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr . A hostPrefix value of 23 provides 510 ($2^{(32 - 23)} - 2$) pod IP addresses.	A subnet prefix. The default value is 23 .
networking.serviceNetwork	The IP address block for services. The default value is 172.30.0.0/16 . The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


Parameter	Description	Values
networking.machineNetwork.cidr	Required if you use networking.machineNetwork . An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24 .	An IP network block in CIDR notation. For example, 10.0.0.0/16 .  NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.


24.5.13.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 24.51. Optional parameters



Parameter	Description	Values
additionalTrustBundle	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String
capabilities	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in <i>Installing</i> .	String array
capabilities.baselineCapabilitySet	Selects an initial set of optional capabilities to enable. Valid values are None , v4.11 , v4.12 and vCurrent . The default value is vCurrent .	String
capabilities.additionalEnabledCapabilities	Extends the set of optional capabilities beyond what you specify in baselineCapabilitySet . You may specify multiple capabilities in this parameter.	String array
compute	The configuration for the machines that comprise the compute nodes.	Array of MachinePool objects.

Parameter	Description	Values
compute.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
compute.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
compute.name	Required if you use compute . The name of the machine pool.	worker
compute.platform	Required if you use compute . Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere , or {}
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to 2 . The default value is 3 .
featureSet	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as TechPreviewNoUpgrade .

Parameter	Description	Values
controlPlane	The configuration for the machines that comprise the control plane.	Array of MachinePool objects.
controlPlane.architecture	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are amd64 (the default).	String
controlPlane.hyperthreading	<p>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</p> <div style="display: flex; align-items: center;">  <div> <p>IMPORTANT</p> <p>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</p> </div> </div>	Enabled or Disabled
controlPlane.name	Required if you use controlPlane . The name of the machine pool.	master
controlPlane.platform	Required if you use controlPlane . Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.	alibabacloud, aws, azure, gcp, ibmcloud, nutanix, openstack, ovirt, vsphere, or {}
controlPlane.replicas	The number of control plane machines to provision.	The only supported value is 3 , which is the default value.

Parameter	Description	Values
credentialsMode	<p>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</p> <div data-bbox="485 510 595 891" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p>NOTE</p> <p>Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the <i>Cloud Credential Operator</i> entry in the <i>Cluster Operators reference</i> content.</p> </div> <div data-bbox="485 936 595 1285" style="border: 1px solid black; padding: 5px;"> <p>NOTE</p> <p>If your AWS account has service control policies (SCP) enabled, you must configure the credentialsMode parameter to Mint, Passthrough or Manual.</p> </div>	Mint, Passthrough, Manual or an empty string ("").

Parameter	Description	Values
fips	<p>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</p> <div data-bbox="488 586 592 1393" style="background-color: black; width: 65px; height: 360px; margin-bottom: 10px;"></div> <p>IMPORTANT</p> <p>To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode. The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64, ppc64le, and s390x architectures.</p> <div data-bbox="488 1442 592 1637" style="background-color: black; width: 65px; height: 87px; margin-bottom: 10px;"></div> <p>NOTE</p> <p>If you are using Azure File storage, you cannot enable FIPS mode.</p>	false or true
imageContentSources	Sources and repositories for the release-image content.	Array of objects. Includes a source and, optionally, mirrors , as described in the following rows of this table.
imageContentSources.source	Required if you use imageContentSources . Specify the repository that users refer to, for example, in image pull specifications.	String

Parameter	Description	Values
imageContentSources.mirrors	Specify one or more repositories that may also contain the same images.	Array of strings
publish	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	<p>Internal or External. The default value is External.</p> <p>Setting this field to Internal is not supported on non-cloud platforms.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>IMPORTANT</p> <p>If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.</p> </div> </div>
sshKey	<p>The SSH key to authenticate access to your cluster machines.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</p> </div> </div>	For example, sshKey: ssh-ed25519 AAAA..

24.5.13.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table.





NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.52. Additional VMware vSphere cluster parameters

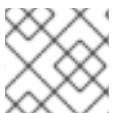
Parameter	Description	Values
vCenter	The fully-qualified hostname or IP address of the vCenter server.	String

Parameter	Description	Values
username	The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.	String
password	The password for the vCenter user name.	String
datacenter	The name of the data center to use in the vCenter instance.	String
defaultDatastore	The name of the default datastore to use for provisioning volumes.	String
folder	Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the data center virtual machine folder.	String, for example, <code>/<datacenter_name>/vm/<folder_name>/<subfolder_name></code> .
resourcePool	Optional. The absolute path of an existing resource pool where the installation program creates the virtual machines. If you do not specify a value, the installation program installs the resources in the root of the cluster under <code>/<datacenter_name>/host/<cluster_name>/Resources</code> .	String, for example, <code>/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name></code> .
network	The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.	String
cluster	The vCenter cluster to install the OpenShift Container Platform cluster in.	String
apiVIPs	The virtual IP (VIP) address that you configured for control plane API access.  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the apiVIP configuration setting is deprecated. Instead, use a List format to enter a value in the apiVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .

Parameter	Description	Values
ingressVIPs	<p>The virtual IP (VIP) address that you configured for cluster ingress.</p>  <p>NOTE</p> <p>In OpenShift Container Platform 4.12 and later, the ingressVIP configuration setting is deprecated. Instead, use a List format to enter a value in the ingressVIPs configuration setting.</p>	An IP address, for example 128.0.0.1 .
diskType	Optional. The disk provisioning method. This value defaults to the vSphere default storage policy if not set.	Valid values are thin , thick , or eagerZeroedThick .

24.5.13.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.53. Optional VMware vSphere machine pool parameters

Parameter	Description	Values
clusterOSImage	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, https://mirror.openshift.com/images/rhcos-<version>-vmware-<architecture>.ova .
osDisk.diskSizeGB	The size of the disk in gigabytes.	Integer
cpus	The total number of virtual processor cores to assign a virtual machine. The value of platform.vsphere.cpus must be a multiple of platform.vsphere.coresPerSocket value.	Integer

Parameter	Description	Values
coresPerSocket	The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket . The default value for control plane nodes and worker nodes is 4 and 2 , respectively.	Integer
memoryMB	The size of a virtual machine's memory in megabytes.	Integer

24.5.13.1.6. Region and zone enablement configuration parameters

To use the region and zone enablement feature, you must specify region and zone enablement parameters in your installation file.



IMPORTANT

Before you modify the **install-config.yaml** file to configure a region and zone enablement environment, read the "VMware vSphere region and zone enablement" and the "Configuring regions and zones for a VMware vCenter" sections.



NOTE

The **platform.vsphere** parameter prefixes each parameter listed in the table.

Table 24.54. Region and zone enablement parameters

Parameter	Description	Values
failureDomains	Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.	String
failureDomains.name	The name of the failure domain. The machine pools use this name to reference the failure domain.	String
failureDomains.server	Specifies the fully-qualified hostname or IP address of the VMware vCenter server, so that a client can access failure domain resources. You must apply the server role to the vSphere vCenter server location.	String
failureDomains.region	You define a region by using a tag from the openshift-region tag category. The tag must be attached to the vCenter datacenter.	String

Parameter	Description	Values
failureDomains.zone	You define a zone by using a tag from the openshift-zone tag category. The tag must be attached to the vCenter datacenter.	String
failureDomains.topology.computeCluster	This parameter defines the compute cluster associated with the failure domain. If you do not define this parameter in your configuration, the compute cluster takes the value of platform.vsphere.cluster and platform.vsphere.datacenter .	String
failureDomains.topology.folder	The absolute path of an existing folder where the installation program creates the virtual machines. If you do not define this parameter in your configuration, the folder takes the value of platform.vsphere.folder .	String
failureDomains.topology.datacenter	Defines the datacenter where OpenShift Container Platform virtual machines (VMs) operate. If you do not define this parameter in your configuration, the datacenter defaults to platform.vsphere.datacenter .	String
failureDomains.topology.datastore	Specifies the path to a vSphere datastore that stores virtual machines files for a failure domain. You must apply the datastore role to the vSphere vCenter datastore location.	String
failureDomains.topology.networks	Lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter in your configuration, the network takes the value of platform.vsphere.network .	String
failureDomains.topology.resourcePool	Optional: The absolute path of an existing resource pool where the installation program creates the virtual machines, for example, /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name> . If you do not specify a value, the installation program installs the resources in the root of the cluster under /<datacenter_name>/host/<cluster_name>/Resources .	String

24.5.13.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster


```

- <mirror_host_name>:<mirror_port>/<repo_name>/release
source: <source_image_1>
- mirrors:
- <mirror_host_name>:<mirror_port>/<repo_name>/release-images
source: <source_image_2>

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 5 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.
- 6 The cluster name that you specified in your DNS records.
- 7 Optional: Provide an existing resource pool for machine creation. If you do not specify a value, the installation program uses the root resource pool of the vSphere cluster.
- 8 The vSphere disk provisioning method.
- 9 The vSphere cluster to install the OpenShift Container Platform cluster in.
- 10 The location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that is accessible from the bastion server.
- 11 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.
- 12 Provide the contents of the certificate file that you used for your mirror registry.
- 13 Provide the **imageContentSources** section from the output of the command to mirror the repository.



NOTE

In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

24.5.13.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

- Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- A proxy URL to use for creating HTTPS connections outside the cluster.
- A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle**

config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

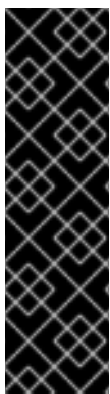


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

24.5.13.4. Configuring regions and zones for a VMware vCenter

You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

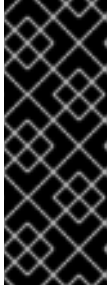


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.

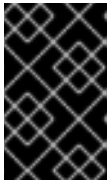


NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```

apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade ❶
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: ❷
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: ❸
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> ❹
    username: <username> ❺
    password: <password> ❻
    datacenter: datacenter ❼
    defaultDatastore: datastore ❽
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" ❾
    cluster: cluster ❿
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" ❶❶
    diskType: thin
    failureDomains: ❶❷
      - name: <machine_pool_zone_1> ❶❸
        region: <region_tag_1> ❶❹
        zone: <zone_tag_1> ❶❺
        topology: ❶❻
          datacenter: <datacenter1> ❶❼
          computeCluster: "/<datacenter1>/host/<cluster1>" ❶❶❶
          resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" ❶❶❷
          networks: ❶❶❸
            - <VM_Network1_name>
          datastore: "/<datacenter1>/datastore/<datastore1>" ❶❶❹
      - name: <machine_pool_zone_2>
        region: <region_tag_2>
        zone: <zone_tag_2>
        topology:
          datacenter: <datacenter2>
          computeCluster: "/<datacenter2>/host/<cluster2>"
          networks:
            - <VM_Network2_name>
          datastore: "/<datacenter2>/datastore/<datastore2>"

```

```
resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
folder: "/<datacenter2>/vm/<folder2>"
# ...
```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**. If you do not specify a value, resources are installed in the root of the cluster **/example_datacenter/host/example_cluster/Resources**.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.5.14. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

When you have configured your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host that is co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Verify the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
--log-level=info 2
```

- 1** For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
- 2** To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



IMPORTANT

Use the **openshift-install** command from the bastion hosted in the VMC environment.



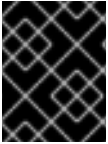
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/openshift_install.log**.



IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

24.5.15. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

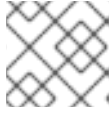
```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.

**NOTE**

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

24.5.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.5.17. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

24.5.18. Creating registry storage

After you install the cluster, you must create storage for the Registry Operator.

24.5.18.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

24.5.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

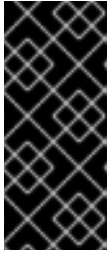
24.5.18.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.

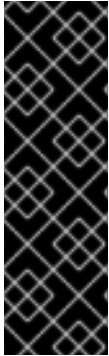
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
■
```

```
storage:
  pvc:
    claim: 1
```

- 1 Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

24.5.19. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.5.20. Services for an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Configuring an external load balancer depends on your vendor's load balancer.

The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

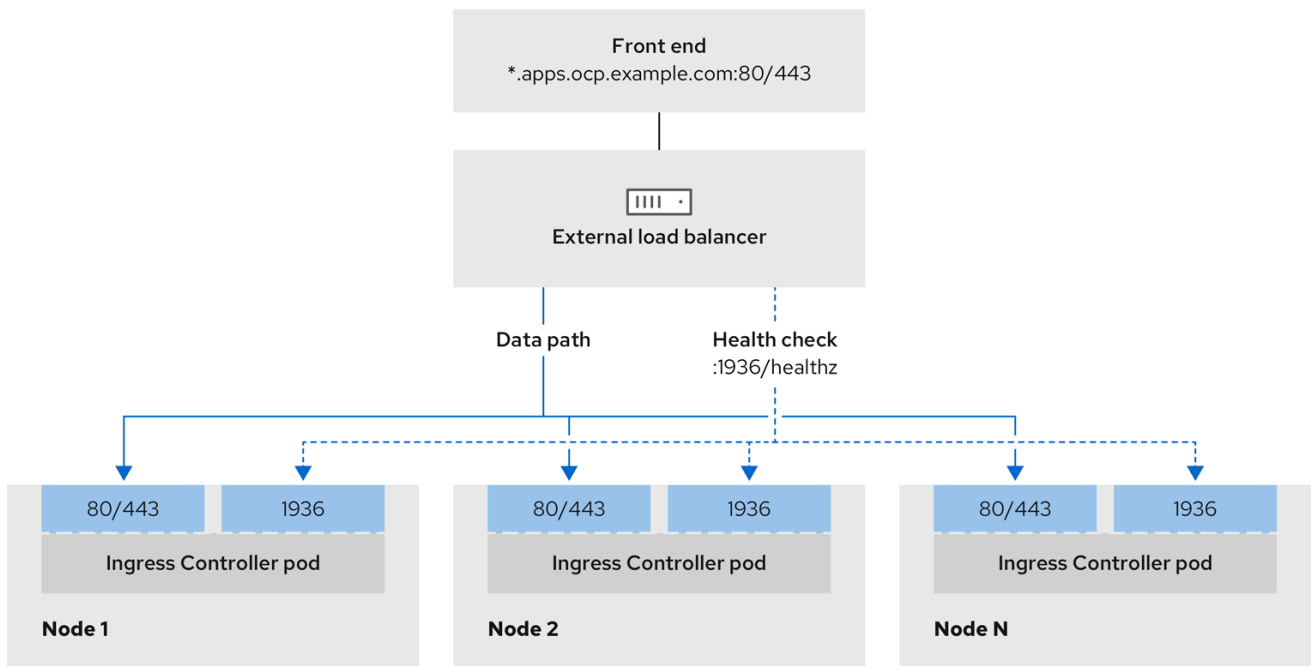
Red Hat supports the following services for an external load balancer:

- Ingress Controller

- OpenShift API
- OpenShift MachineConfig API

You can choose whether you want to configure one or all of these services for an external load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 24.10. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment



496_OpenShift_1223

Figure 24.11. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment

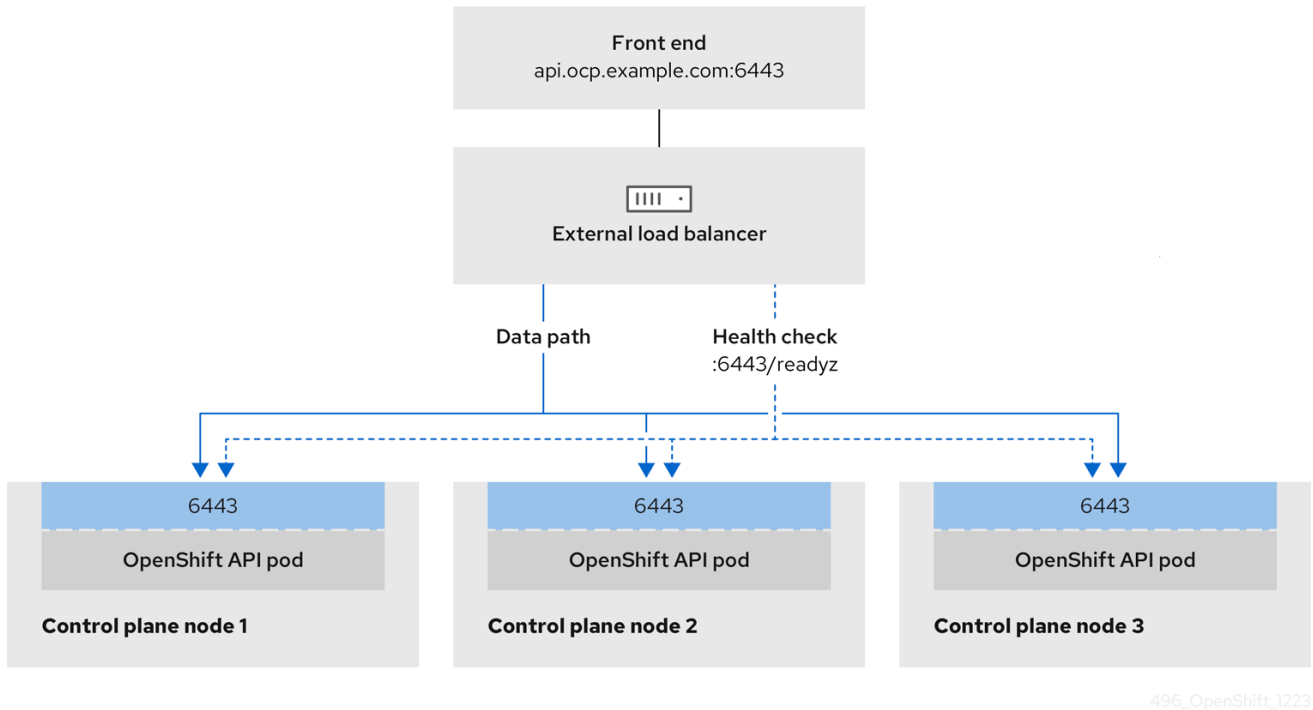
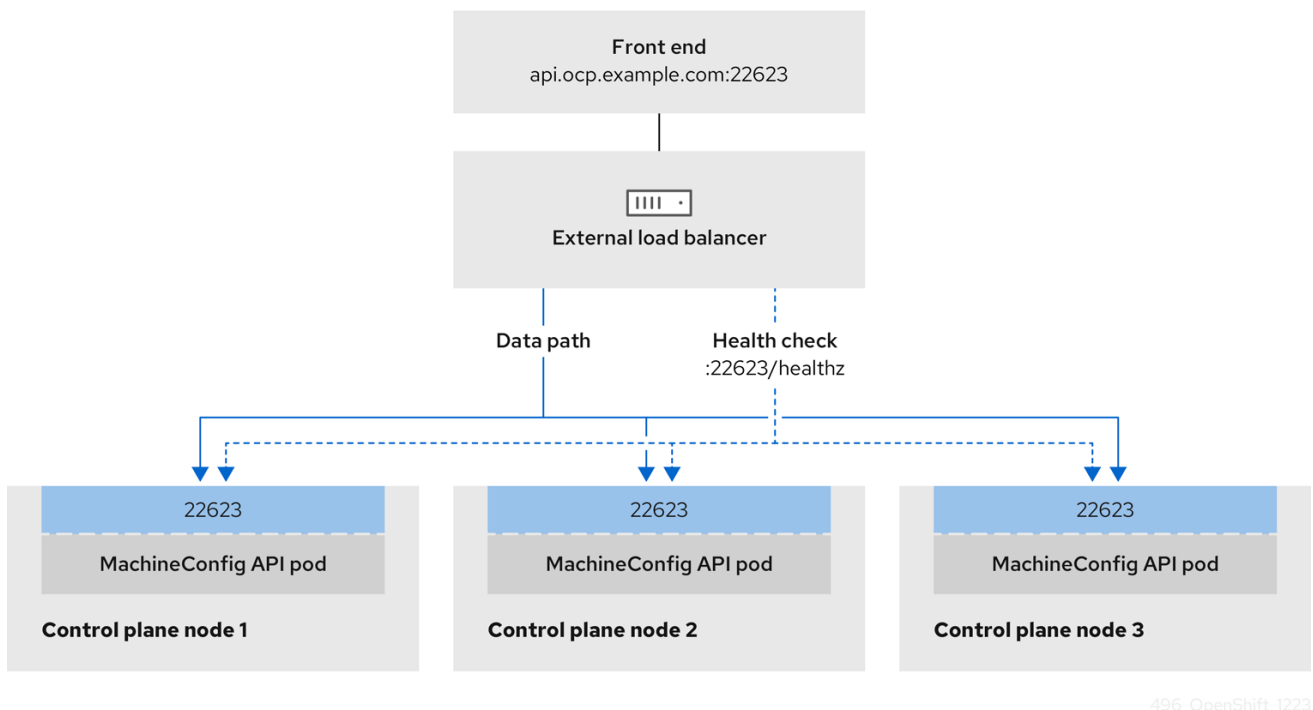


Figure 24.12. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for external load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.

- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a `/27` or `/28`, you can simplify your load balancer targets.

TIP

You can list all IP addresses that exist in a network by checking the machine config pool's resources.

Before you configure an external load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the external load balancer. You can achieve this by completing one of the following actions:
 - Assign a static IP address to each control plane node.
 - Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the external load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

24.5.20.1. Configuring an external load balancer

You can configure an OpenShift Container Platform cluster to use an external load balancer in place of the default load balancer.



IMPORTANT

Before you configure an external load balancer, ensure that you read the "Services for an external load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your external load balancer.



NOTE

MetalLB, that runs on a cluster, functions as an external load balancer.

OpenShift API prerequisites

- You defined a front-end IP address.
- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:

- Port 6443 provides access to the OpenShift API service.
- Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

Ingress Controller prerequisites

- You defined a front-end IP address.
- TCP ports 443 and 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are be reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address, port 80 and port 443 are reachable to all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

Prerequisite for health check URL specifications

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following examples demonstrate health check specifications for the previously listed backend services:

Example of a Kubernetes API health check specification

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of a Machine Config API health check specification

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Example of an Ingress Controller health check specification

```

Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10

```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 443, and 80:

Example HAProxy configuration

```

#...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
    option httpchk
    http-check connect
    http-check send meth GET uri /readyz
    http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
    option httpchk
    http-check connect
    http-check send meth GET uri /healthz
    http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
    option httpchk
    http-check connect
    http-check send meth GET uri /healthz/ready
    http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
    bind 192.168.1.100:80
    mode tcp
    balance roundrobin
    option httpchk

```

```

http-check connect
http-check send meth GET uri /healthz/ready
http-check expect status 200
  server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
  server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...

```

2. Use the **curl** CLI command to verify that the external load balancer and its resources are operational:

- a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```

{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}

```

- b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 200 OK
Content-Length: 0

```

- c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

If the configuration is correct, the output from the command shows the following response:

```

HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache

```

- d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.  
<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-  
console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK  
referrer-policy: strict-origin-when-cross-origin  
set-cookie: csrf-  
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEJhJfYqWwCGBsja261dG  
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax  
x-content-type-options: nosniff  
x-dns-prefetch-control: off  
x-frame-options: DENY  
x-xss-protection: 1; mode=block  
date: Wed, 04 Oct 2023 16:29:38 GMT  
content-type: text/html; charset=utf-8  
set-cookie:  
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;  
HttpOnly; Secure; SameSite=None  
cache-control: private
```

3. Configure the DNS records for your cluster to target the front-end IP addresses of the external load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer.

Examples of modified DNS records

```
<load_balancer_ip_address> A api.<cluster_name>.<base_domain>  
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address> A apps.<cluster_name>.<base_domain>  
A record pointing to Load Balancer Front End
```



IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

4. Use the **curl** CLI command to verify that the external load balancer and DNS record configuration are operational:
 - a. Verify that you can access the cluster API, by running the following command and observing the output:

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

If the configuration is correct, you receive a JSON object in response:

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

- b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

- c. Verify that you can access each cluster application on port, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXIfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

- d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UIYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJfYqWwCGBsja261dG
LgaYO0nxzVERhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

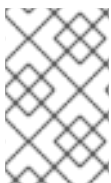
24.5.21. Next steps

- [Customize your cluster.](#)
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#) .
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .

24.6. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure that you provision by deploying it to [VMware Cloud \(VMC\) on AWS](#) .

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

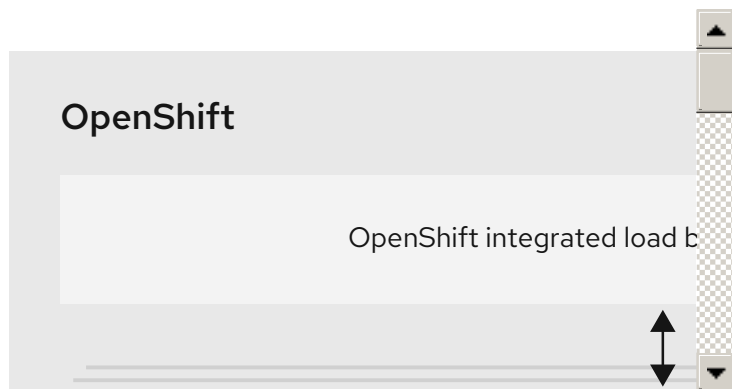


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.6.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the internet. This is used by nodes and applications to download container images.
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**

**NOTE**

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.6.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

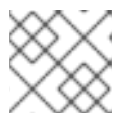
To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.6.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update processes](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

24.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.6.4. VMware vSphere infrastructure requirements

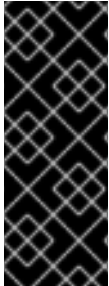
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.55. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.56. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.6.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later

- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#).
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.6.6. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

24.6.6.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 24.13. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API
		VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.Add

vSphere object for role	delete privileges are optional if your cluster does not use the Machine API. When required	NewDisk Required privileges in vSphere API VirtualMachine.Config.AddNewDevice
		VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.14. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine".Change Configuration".Change resource" "Virtual machine".Change Configuration".Change Settings" "Virtual machine".Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine".Edit Inventory".Create new" "Virtual machine".Edit Inventory".Create from existing" "Virtual machine".Edit Inventory".Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine".Change Configuration".Add existing disk" "Virtual machine".Change Configuration".Add new disk" "Virtual machine".Change

vSphere object for role	When required	Configuration". "Add or Remove device" GUI "Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from existing virtual machine". "Edit virtual machine". "Remove virtual machine". "Clone virtual machine". "Deploy template". "Mark as template". "Create folder". "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.15. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

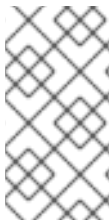
If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 24.57. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

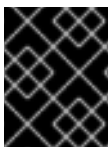
- [Creating a compute machine set on vSphere](#)

24.6.6.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 24.58. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

24.6.6.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 24.59. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

24.6.6.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

24.6.6.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow

the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

24.6.6.5.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

24.6.6.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 24.60. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.61. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.62. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00** to **00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00** to **00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00** to **00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00** to **00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

24.6.6.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

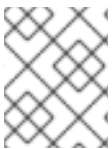
It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 24.63. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

24.6.6.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 24.16. Sample DNS zone database

```

$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

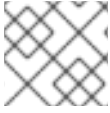
Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 24.17. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

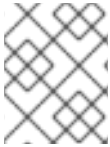
- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

24.6.6.7. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 24.64. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 24.65. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

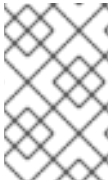
**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

24.6.6.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 24.18. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode            http
  log             global
  option          dontlognull
  option http-server-close
  option          redispatch
  retries         3
  timeout http-request 10s
  timeout queue    1m
  timeout connect  10s
  timeout client   1m
  timeout server   1m
  timeout http-keep-alive 10s
  timeout check    10s
  maxconn         3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5

```

```

bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

24.6.7. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

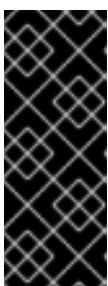
- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.

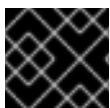
- a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

24.6.8. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

24.6.9. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

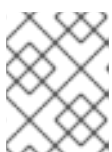
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

■

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

24.6.10. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

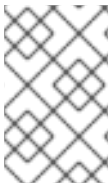
The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify

multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.66. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.6.11. Obtaining the installation program

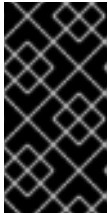
Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

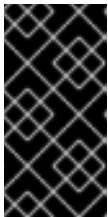
Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

24.6.12. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

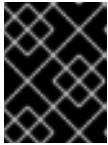
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

24.6.12.1. Sample **install-config.yaml** file for VMware vSphere

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12
```

```
resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 13
diskType: thin 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, (-), and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 3 You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.
- 5 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 6 The cluster name that you specified in your DNS records.
- 7 The fully-qualified hostname or IP address of the vCenter server.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

- 8 The name of the user for accessing the server.
- 9 The password associated with the vSphere user.
- 10 The vSphere datacenter.
- 11 The default vSphere datastore to use.
- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster and you do not want to use the default **StorageClass** object, named **thin**, you can omit the **folder** parameter from the **install-config.yaml** file.
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the

installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 The pull secret that you obtained from [OpenShift Cluster Manager Hybrid Cloud Console](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

24.6.12.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

**NOTE**

The installation program does not support the proxy **readinessEndpoints** field.

**NOTE**

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

24.6.12.3. Configuring regions and zones for a VMware vCenter

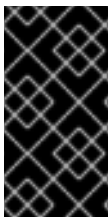
You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

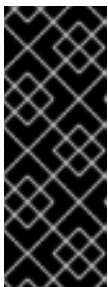


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
vsphere:
  zones: 2
  - "<machine_pool_zone_1>"
  - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
vsphere:
  zones: 3
```



```

- "<machine_pool_zone_1>"
- "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> 4
    username: <username> 5
    password: <password> 6
    datacenter: datacenter 7
    defaultDatastore: datastore 8
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
    cluster: cluster 10
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
    diskType: thin
    failureDomains: 12
    - name: <machine_pool_zone_1> 13
      region: <region_tag_1> 14
      zone: <zone_tag_1> 15
      topology: 16
        datacenter: <datacenter1> 17
        computeCluster: "/<datacenter1>/host/<cluster1>" 18
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
        networks: 20
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>" 21
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.

- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.6.13. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
3. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.

- To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

24.6.14. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- The output of this command is your cluster name and a random string.

24.6.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

- 1** Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

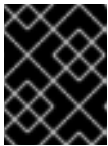
3. Locate the following Ignition config files that the installation program created:
 - **<installation_directory>/master.ign**

- `<installation_directory>/worker.ign`
 - `<installation_directory>/merge-bootstrap.ign`
4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

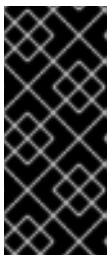
```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.

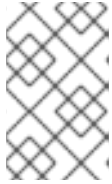


IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
- Click the **VMs and Templates** view.
 - Right-click the name of your datacenter.
 - Click **New Folder** → **New VM and Template Folder**
 - In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
 - Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
- g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

**IMPORTANT**

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.

**IMPORTANT**

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone** → **Clone to Virtual Machine**

- b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.



IMPORTANT

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```

- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded ignition config file for this machine type.

encoded ignition config file for this machine type.

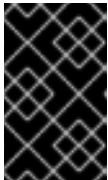
- **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power** → **Power On**.
- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

24.6.16. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone** → **Clone to Virtual Machine**

2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.

**NOTE**

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**.
7. On the **Customize hardware** tab, click **Advanced**.
 - a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

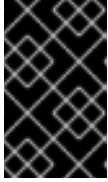
- Continue to create more compute machines for your cluster.

24.6.17. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions**: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.

**IMPORTANT**

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.

**IMPORTANT**

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

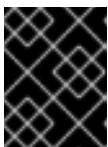
- **Retain existing partitions:** For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate /var partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers:** Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd:** Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var:** Holds data that you might want to keep separate for purposes such as auditing.

**IMPORTANT**

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

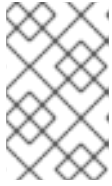
- Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

- Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

24.6.18. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your **PATH**.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

24.6.19. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

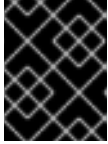
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

24.6.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.6.21. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.25.0
master-1	Ready	master	63m	v1.25.0
master-2	Ready	master	64m	v1.25.0

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-8b2br	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
csr-8vnps	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
...			

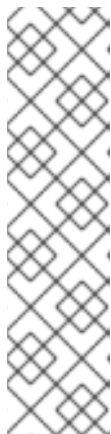
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

24.6.22. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m

console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

24.6.22.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

24.6.22.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.6.22.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

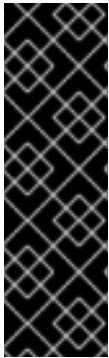
- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	False

24.6.22.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

24.6.22.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

- a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage ①
  namespace: openshift-image-registry ②
spec:
  accessModes:
  - ReadWriteOnce ③
  resources:
    requests:
      storage: 100Gi ④
```

- ① A unique name that represents the **PersistentVolumeClaim** object.
- ② The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- ③ The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- ④ The size of the persistent volume claim.

- b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

- Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

24.6.23. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

- Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m

kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.
 - a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE              NAME                                READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8      1/1
Running   0    5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

24.6.24. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.

4. Create a backup of the cloned volume.
5. Delete the cloned volume.

24.6.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.6.26. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- [Set up your registry and configure registry storage](#).
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.7. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE AND NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.12, you can install a cluster on your VMware vSphere instance using infrastructure you provision with customized network configuration options by deploying it to [VMware Cloud \(VMC\) on AWS](#).

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing VXLAN configurations. You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

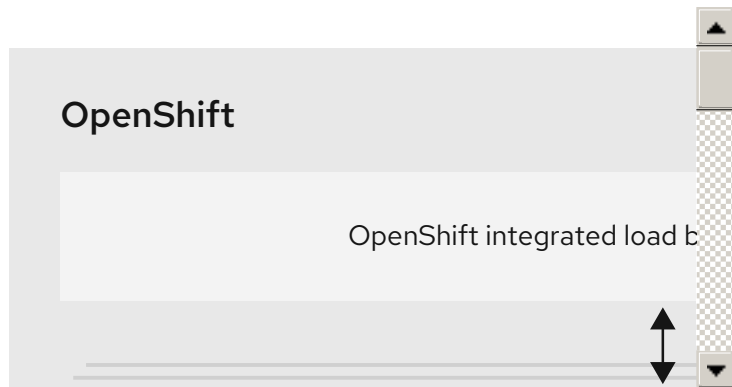


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.7.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the internet. This is used by nodes and applications to download container images.
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
 - The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**

- Network name
- Datastore name, such as **WorkloadDatastore**

**NOTE**

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.7.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM

- Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.7.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update processes](#).
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.

24.7.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.7.4. VMware vSphere infrastructure requirements

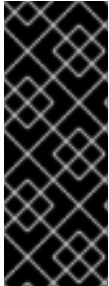
You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.67. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later

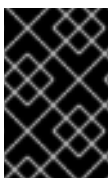


IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.68. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

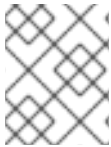
24.7.5. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later

- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.7.6. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

24.7.6.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 24.19. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.Add

vSphere object for role	When required	RemoveDevice Required privileges in vSphere API VirtualMachine.Config.AdvancedConfig
		VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.Add

vSphere object for role	delete privileges are optional if your cluster does not use the Machine API. When required	NewDisk Required privileges in vSphere API VirtualMachine.Config.AddRemoveDevice
		VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.20. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine".Change GUI Configuration".Reset guest information"
		"Virtual machine"."Change Configuration".Change resource" "Virtual machine"."Change Configuration".Change Settings" "Virtual machine"."Change Configuration".Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration".Add existing disk" "Virtual machine"."Change Configuration".Add new disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Add or Remove device" GUI "Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from Existing Virtual machine". "Edit Virtual machine". "Remove Virtual machine". "Provisioning. "Clone virtual machine" "Virtual machine". "Provisioning. "Deploy template" "Virtual machine". "Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.21. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 24.69. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

- [Creating a compute machine set on vSphere](#)

24.7.6.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 24.70. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

24.7.6.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 24.71. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

24.7.6.4. Certificate signing requests management

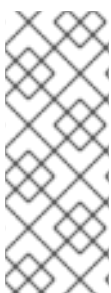
Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

24.7.6.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow

the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

24.7.6.5.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

24.7.6.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 24.72. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.73. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.74. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00 to 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 to 00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

24.7.6.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 24.75. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <h4>IMPORTANT</h4> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

24.7.6.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 24.22. Sample DNS zone database

```

$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

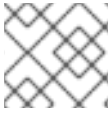
Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 24.23. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

24.7.6.7. Load balancing requirements for user-provisioned infrastructure

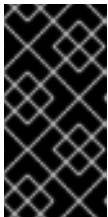
Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 24.76. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 24.77. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

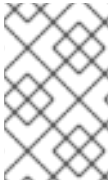
**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

24.7.6.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 24.24. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode            http
  log             global
  option          dontlognull
  option http-server-close
  option          redispatch
  retries         3
  timeout http-request 10s
  timeout queue    1m
  timeout connect  10s
  timeout client   1m
  timeout server   1m
  timeout http-keep-alive 10s
  timeout check    10s
  maxconn         3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5

```

```

bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

24.7.7. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.

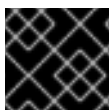
- a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

24.7.8. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

**NOTE**

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

24.7.9. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

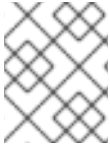
After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

-

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

24.7.10. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default **install-config.yaml** file includes **vcenters** and **failureDomains** fields, where you can specify multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave

these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.78. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.7.11. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.



IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.



IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#) . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

24.7.12. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

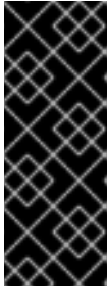
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

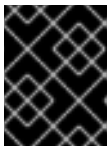
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

24.7.12.1. Sample **install-config.yaml** file for VMware vSphere

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12
```

```
resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 13
diskType: thin 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, (-), and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 3 You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.
- 5 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 6 The cluster name that you specified in your DNS records.
- 7 The fully-qualified hostname or IP address of the vCenter server.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

- 8 The name of the user for accessing the server.
- 9 The password associated with the vSphere user.
- 10 The vSphere datacenter.
- 11 The default vSphere datastore to use.
- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster and you do not want to use the default **StorageClass** object, named **thin**, you can omit the **folder** parameter from the **install-config.yaml** file.
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the

installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 The pull secret that you obtained from [OpenShift Cluster Manager Hybrid Cloud Console](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

24.7.12.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```

apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.



NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

24.7.12.3. Configuring regions and zones for a VMware vCenter

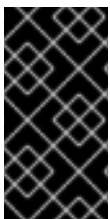
You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

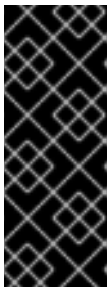


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

1. Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
vsphere:
  zones: 2
  - "<machine_pool_zone_1>"
  - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
vsphere:
  zones: 3
```

```

- "<machine_pool_zone_1>"
- "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> 4
    username: <username> 5
    password: <password> 6
    datacenter: datacenter 7
    defaultDatastore: datastore 8
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
    cluster: cluster 10
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
    diskType: thin
    failureDomains: 12
    - name: <machine_pool_zone_1> 13
      region: <region_tag_1> 14
      zone: <zone_tag_1> 15
      topology: 16
        datacenter: <datacenter1> 17
        computeCluster: "/<datacenter1>/host/<cluster1>" 18
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
        networks: 20
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>" 21
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
# ...

```

- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.

- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.
- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.7.13. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment. You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.
2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following examples:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig: {}
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.
5. Remove the Kubernetes manifest files that define the control plane machines and compute machineSets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the MachineSet files to create compute machines by using the machine API, but you must update references to them to match your environment.

24.7.14. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group and these fields cannot be changed:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

24.7.14.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 24.79. Cluster Network Operator configuration object


Field	Type	Description
metadata.name	string	The name of the CNO object. This name is always cluster .
spec.clusterNetwork	array	<p>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</p> <pre>spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>

Field	Type	Description
spec.serviceNetwork	array	<p>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes network plugins support only a single IP address block for the service network. For example:</p> <pre>spec: serviceNetwork: - 172.30.0.0/14</pre> <p>You can customize this field only in the install-config.yaml file before you create the manifests. The value is read-only in the manifest file.</p>
spec.defaultNetwork	object	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	object	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 24.80. **defaultNetwork** object

Field	Type	Description
type	string	<p>Either OpenShiftSDN or OVNKubernetes. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.</p> </div> </div>
openshiftSDNConfig	object	This object is only valid for the OpenShift SDN network plugin.
ovnKubernetesConfig	object	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OpenShift SDN network plugin

The following table describes the configuration fields for the OpenShift SDN network plugin:

Table 24.81. **openshiftSDNConfig** object

Field	Type	Description
mode	string	<p>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</p> <p>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</p>
mtu	integer	<p>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.</p> <p>This value cannot be changed after cluster installation.</p>
vxlanPort	integer	<p>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.</p> <p>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</p> <p>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</p>


Example OpenShift SDN configuration

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 24.82. `ovnKubernetesConfig` object

Field	Type	Description
mtu	integer	<p>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</p> <p>If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</p> <p>If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.</p>
genevePort	integer	The port to use for all Geneve packets. The default value is 6081 . This value cannot be changed after cluster installation.
ipsecConfig	object	Specify an empty object to enable IPsec encryption.
policyAuditConfig	object	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	object	<p>Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  </div> <div style="flex: 2;"> <p>NOTE</p> <p>While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.</p> </div> </div>

Field	Type	Description
v4InternalSubnet	<p>If your existing network infrastructure overlaps with the 100.64.0.0/16 IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the clusterNetwork.cidr value is 10.128.0.0/14 and the clusterNetwork.hostPrefix value is /23, then the maximum number of nodes is $2^{(23-14)}=512$.</p> <p>This field cannot be changed after installation.</p>	<p>The default value is 100.64.0.0/16.</p>

Field	Type	Description
v6InternalSubnet	<p>If your existing network infrastructure overlaps with the fd98::/48 IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster.</p> <p>This field cannot be changed after installation.</p>	The default value is fd98::/48 .

Table 24.83. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is 20 messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is 50000000 or 50 MB.

Field	Type	Description
destination	string	<p>One of the following additional audit log targets:</p> <p>libc The libc syslog() function of the journald process on the host.</p> <p>udp:<host>:<port> A syslog server. Replace <host>:<port> with the host and port of the syslog server.</p> <p>unix:<file> A Unix Domain Socket file specified by <file>.</p> <p>null Do not send the audit logs to any additional target.</p>
syslogFacility	string	The syslog facility, such as kern , as defined by RFC5424. The default value is local0 .

Table 24.84. gatewayConfig object

Field	Type	Description
routingViaHost	boolean	<p>Set this field to true to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is false.</p> <p>This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to true, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.</p>


Example OVN-Kubernetes configuration with IPsec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig: {}
```

kubeProxyConfig object configuration

The values for the **kubeProxyConfig** object are defined in the following table:

Table 24.85. kubeProxyConfig object

Field	Type	Description
iptablesSyncPeriod	string	<p>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>NOTE</p> <p>Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.</p> </div> </div>
proxyArguments.iptables-min-sync-period	array	<p>The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:</p> <pre>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</pre>

24.7.15. Creating the Ignition config files

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

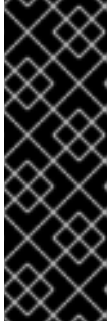
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

Procedure

- Obtain the Ignition config files:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.



IMPORTANT

If you created an **install-config.yaml** file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

The following files are generated in the directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

24.7.16. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```


- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

24.7.17. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {}
}
```

```
"storage": {},
"systemd": {}
}
```

- 1 Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

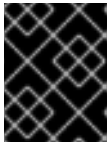
- **<installation_directory>/master.ign**
- **<installation_directory>/worker.ign**
- **<installation_directory>/merge-bootstrap.ign**

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

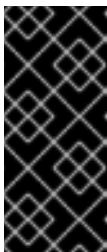
6. In the vSphere Client, create a folder in your datacenter to store your VMs.
 - a. Click the **VMs and Templates** view.
 - b. Right-click the name of your datacenter.

- c. Click **New Folder → New VM and Template Folder**.
 - d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

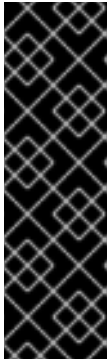
In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
 - Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
- g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

**IMPORTANT**

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.



IMPORTANT

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone → Clone to Virtual Machine**
 - b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.



IMPORTANT

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```

- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.
- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

24.7.18. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone → Clone to Virtual Machine**
2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**
7. On the **Customize hardware** tab, click **Advanced**.
 - a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

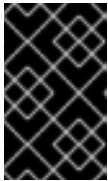
- Continue to create more compute machines for your cluster.

24.7.19. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

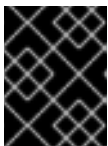
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions:** For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.



IMPORTANT

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

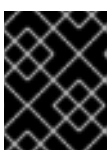
- **Retain existing partitions:** For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate **/var** partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers:** Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd:** Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var:** Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this

method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

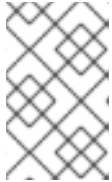
```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
      partitions:
        - label: var
          start_mib: <partition_start_offset> 2
          size_mib: <partition_size> 3
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] 4
          with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.

- 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

24.7.20. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

24.7.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.7.22. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready   master   63m   v1.25.0
master-1  Ready   master   63m   v1.25.0
master-2  Ready   master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps  15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrap** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}\n\n{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

NAME	AGE	REQUESTOR	CONDITION
csr-bfd72	5m26s	system:node:ip-10-0-50-126.us-east-2.compute.internal	Pending
csr-c57lv	5m26s	system:node:ip-10-0-95-157.us-east-2.compute.internal	Pending
...			

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

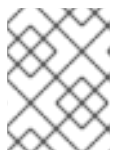
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	73m	v1.25.0
master-1	Ready	master	73m	v1.25.0
master-2	Ready	master	74m	v1.25.0
worker-0	Ready	worker	11m	v1.25.0
worker-1	Ready	worker	11m	v1.25.0



NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

24.7.23. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

24.7.23.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

24.7.23.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.7.23.2.1. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.

4 The size of the persistent volume claim.

b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

24.7.24. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m

dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

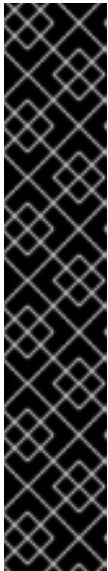
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.



IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE           NAME                                     READY  STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running   1    9m
openshift-apiserver          apiserver-67b9g                                1/1  Running  0
3m
openshift-apiserver          apiserver-ljcmx                                1/1  Running  0
1m
openshift-apiserver          apiserver-z25h4                                1/1  Running  0
2m
openshift-authentication-operator  authentication-operator-69d5d8bf84-vh2n8      1/1
Running   0    5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

24.7.25. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

24.7.26. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

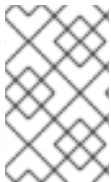
24.7.27. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#).
- [Set up your registry and configure registry storage](#).
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.8. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.12, you can install a cluster on VMware vSphere infrastructure that you provision in a restricted network by deploying it to [VMware Cloud \(VMC\) on AWS](#).

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

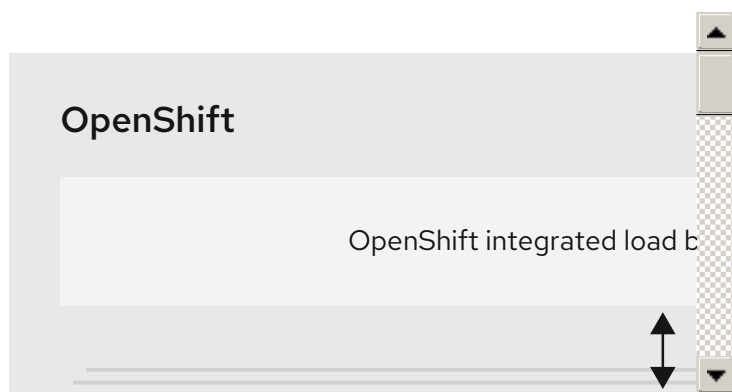


NOTE

OpenShift Container Platform supports deploying a cluster to a single VMware vCenter only. Deploying a cluster with machines/machine sets on multiple vCenters is not supported.

24.8.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.



You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Configure the following firewall rules:
 - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
 - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.
- You must have the following information to deploy OpenShift Container Platform:
 - The OpenShift Container Platform cluster name, such as **vmc-prod-1**.
 - The base DNS name, such as **companyname.com**.
 - If not using the default, the pod network CIDR and services network CIDR must be

identified, which are set by default to **10.128.0.0/14** and **172.30.0.0/16**, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.

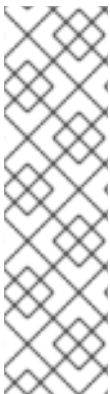
- The following vCenter information:
 - vCenter hostname, username, and password
 - Datacenter name, such as **SDDC-Datacenter**
 - Cluster name, such as **Cluster-1**
 - Network name
 - Datastore name, such as **WorkloadDatastore**



NOTE

It is recommended to move your vSphere cluster to the VMC **Compute-ResourcePool** resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
 - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have internet connectivity and the ability to upload an OVA to the ESXi hosts.
 - Download and install the OpenShift CLI tools to the bastion host.
 - The **openshift-install** installation program
 - The OpenShift CLI (**oc**) tool



NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

24.8.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

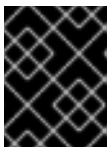
To determine this, VMware provides the [VMC on AWS Sizer](#). With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
 - Storage requirements
 - vCPUs
 - vRAM
 - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

24.8.2. vSphere prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- You [created a registry on your mirror host](#) and obtain the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You provisioned [block registry storage](#). For more information on persistent storage, see [Understanding persistent storage](#).
- If you use a firewall and plan to use the Telemetry service, you [configured the firewall to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

24.8.3. About installations in restricted networks

In OpenShift Container Platform 4.12, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.



IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

24.8.3.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

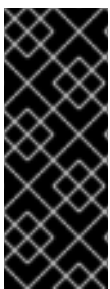
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

24.8.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

24.8.5. VMware vSphere infrastructure requirements

You must install an OpenShift Container Platform cluster on one of the following versions of a VMware vSphere instance that meets the requirements for the components that you use:

- Version 7.0 Update 2 or later
- Version 8.0 Update 1 or later

You can host the VMware vSphere infrastructure on-premise or on a [VMware Cloud Verified provider](#) that meets the requirements outlined in the following table:

Table 24.86. Version requirements for vSphere virtual environments

Virtual environment product	Required version
VMware virtual hardware	15 or later
vSphere ESXi hosts	7.0 Update 2 or later; 8.0 Update 1 or later
vCenter host	7.0 Update 2 or later; 8.0 Update 1 or later



IMPORTANT

Installing a cluster on VMware vSphere versions 7.0 and 7.0 Update 1 is deprecated. These versions are still fully supported, but all vSphere 6.x versions are no longer supported. Version 4.12 of OpenShift Container Platform requires VMware virtual hardware version 15 or later. To update the hardware version for your vSphere virtual machines, see the "Updating hardware on nodes running in vSphere" article in the *Updating clusters* section.

Table 24.87. Minimum supported vSphere version for VMware components

Component	Minimum supported versions	Description
Hypervisor	vSphere 7.0 Update 2 (or later) or vSphere 8.0 Update 1 (or later) with virtual hardware version 15	This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. For more information about supported hardware on the latest version of Red Hat Enterprise Linux (RHEL) that is compatible with RHCOS, see Hardware on the Red Hat Customer Portal.
Storage with in-tree drivers	vSphere 7.0 Update 2 or later; 8.0 Update 1 or later	This plugin creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.



IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

24.8.6. VMware vSphere CSI Driver Operator requirements

To install the vSphere CSI Driver Operator, the following requirements must be met:

- VMware vSphere version: 7.0 Update 2 or later; 8.0 Update 1 or later
- vCenter version: 7.0 Update 2 or later; 8.0 Update 1 or later
- Virtual machines of hardware version 15 or later
- No third-party vSphere CSI driver already installed in the cluster

If a third-party vSphere CSI driver is present in the cluster, OpenShift Container Platform does not overwrite it. The presence of a third-party vSphere CSI driver prevents OpenShift Container Platform from updating to OpenShift Container Platform 4.13 or later.



NOTE

The VMware vSphere CSI Driver Operator is supported only on clusters deployed with **platform: vsphere** in the installation manifest.

Additional resources

- To remove a third-party CSI driver, see [Removing a third-party vSphere CSI Driver](#) .
- To update the hardware version for your vSphere nodes, see [Updating hardware on nodes running in vSphere](#).

24.8.7. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

24.8.7.1. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that you provided, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, your vSphere account must include privileges for reading and creating the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

Example 24.25. Roles and privileges required for installation in vSphere API

vSphere object for role	When required	Required privileges in vSphere API
-------------------------	---------------	------------------------------------

vSphere object for role	When required	Required privileges in vSphere API
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Config.StorageResource.AssignVMToPool VApp.AssignResourcePool VApp.Import VirtualMachine.Config.AddNewDisk
vSphere Datastore	Always	Datastore.AllocateSpace Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable
vSphere Port Group	Always	Network.Assign
Virtual Machine Folder	Always	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk

vSphere object for role	When required	Required privileges in vSphere API
		VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Create and VirtualMachine.Inventory.D	InventoryService.Tagging.ObjectAttachable Resource.AssignVMToPool VApp.Import VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.Add

vSphere object for role	delete privileges are optional if your cluster does not use the Machine API. When required	NewDisk Required privileges in vSphere API VirtualMachine.Config.AddNewDevice
		VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

Example 24.26. Roles and privileges required for installation in vCenter graphical user interface (GUI)

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere vCenter	Always	Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"
vSphere vCenter Cluster	If VMs will be created in the cluster root	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"
vSphere vCenter Resource Pool	If an existing resource pool is provided	Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"

vSphere object for role	When required	Required privileges in vCenter GUI
vSphere Datastore	Always	Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"
vSphere Port Group	Always	Network."Assign network"
Virtual Machine Folder	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration".Rename Virtual machine : Change Configuration".Reset guest GUI information"
		"Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Gues t operating system management by VIX API" "Virtual machine".Interaction."Powe r off" "Virtual machine".Interaction."Powe r on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clo ne virtual machine" "Virtual machine".Provisioning."Mar k as template" "Virtual machine".Provisioning."De ploy template"
vSphere vCenter Datacenter	If the installation program creates the virtual machine folder. For UPI, VirtualMachine.Inventory.Cr eate and VirtualMachine.Inventory.D elete privileges are optional if your cluster does not use the Machine API.	"vSphere Tagging".Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change

vSphere object for role	When required	Configuration". "Add or Remove device" Virtual machine". "Change Configuration". "Advanced configuration" "Virtual machine". "Change Configuration". "Set annotation" "Virtual machine". "Change Configuration". "Change CPU count" "Virtual machine". "Change Configuration". "Extend virtual disk" "Virtual machine". "Change Configuration". "Acquire disk lease" "Virtual machine". "Change Configuration". "Modify device settings" "Virtual machine". "Change Configuration". "Change Memory" "Virtual machine". "Change Configuration". "Remove disk" "Virtual machine". "Change Configuration". Rename "Virtual machine". "Change Configuration". "Reset guest information" "Virtual machine". "Change Configuration". "Change resource" "Virtual machine". "Change Configuration". "Change Settings" "Virtual machine". "Change Configuration". "Upgrade virtual machine compatibility" "Virtual machine". Interaction. "Guest operating system management by VIX API" "Virtual machine". Interaction. "Power off" "Virtual machine". Interaction. "Power on" "Virtual machine". Interaction. Reset "Virtual machine". "Edit Inventory". "Create new" "Virtual machine". "Edit

vSphere object for role	When required	Inventory". "Create from Existing Virtual machine". "Edit Virtual machine". "Remove Virtual machine". Provisioning. "Clone virtual machine" Virtual machine". Provisioning. "Deploy template" Virtual machine". Provisioning. "Mark as template" Folder. "Create folder" Folder. "Delete folder"

Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

Example 24.27. Required permissions and propagation settings

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter	Always	False	Listed required privileges
vSphere vCenter Datacenter	Existing folder	False	ReadOnly permission
	Installation program creates the folder	True	Listed required privileges
vSphere vCenter Cluster	Existing resource pool	False	ReadOnly permission
	VMs in cluster root	True	Listed required privileges
vSphere vCenter Datastore	Always	False	Listed required privileges
vSphere Switch	Always	False	ReadOnly permission
vSphere Port Group	Always	False	Listed required privileges
vSphere vCenter Virtual Machine Folder	Existing folder	True	Listed required privileges

vSphere object	When required	Propagate to children	Permissions required
vSphere vCenter Resource Pool	Existing resource pool	True	Listed required privileges

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

Using OpenShift Container Platform with vMotion

If you intend on using vMotion in your vSphere environment, consider the following before installing an OpenShift Container Platform cluster.

- OpenShift Container Platform generally supports compute-only vMotion, where *generally* implies that you meet all VMware best practices for vMotion. To help ensure the uptime of your compute and control plane nodes, ensure that you follow the VMware best practices for vMotion, and use VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

For more information about vMotion and anti-affinity rules, see the VMware vSphere documentation for [vMotion networking requirements](#) and [VM anti-affinity rules](#).

- Using Storage vMotion can cause issues and is not supported. If you are using vSphere volumes in your pods, migrating a VM across datastores, either manually or through Storage vMotion, causes invalid references within OpenShift Container Platform persistent volume (PV) objects that can result in data loss.
- OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources

When you deploy an OpenShift Container Platform cluster that uses infrastructure that you provided, you must create the following resources in your vCenter instance:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
 - 1 template
 - 1 temporary bootstrap node
 - 3 control plane nodes
 - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements

You must use the Dynamic Host Configuration Protocol (DHCP) for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. In the DHCP lease, you must configure the DHCP to use the default gateway. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:



NOTE

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses

DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**

Table 24.88. Required DNS records

Component	Record	Description
API VIP	api.<cluster_name>.<base_domain>.	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Additional resources

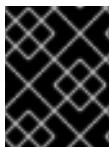
- [Creating a compute machine set on vSphere](#)

24.8.7.2. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 24.89. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

24.8.7.3. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 24.90. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

24.8.7.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

24.8.7.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.



NOTE

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow

the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

24.8.7.5.1. Setting the cluster node hostnames through DHCP

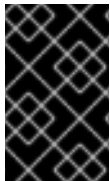
On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

24.8.7.5.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.



IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 24.91. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 24.92. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 24.93. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

Ethernet adaptor hardware address requirements

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- **00:05:69:00:00:00 to 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 to 00:50:56:3F:FF:FF**

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

24.8.7.6. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

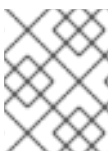
It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 24.94. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

24.8.7.6.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 24.28. Sample DNS zone database

```

$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 24.29. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

24.8.7.7. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 24.95. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 24.96. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

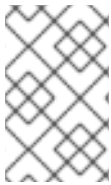
**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

24.8.7.7.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 24.30. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode            http
  log             global
  option          dontlognull
  option http-server-close
  option          redispatch
  retries         3
  timeout http-request 10s
  timeout queue    1m
  timeout connect  10s
  timeout client   1m
  timeout server   1m
  timeout http-keep-alive 10s
  timeout check    10s
  maxconn         3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5

```

```

bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

24.8.8. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.

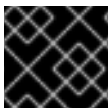
- a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

24.8.9. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

24.8.10. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

■

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

24.8.11. VMware vSphere region and zone enablement

You can deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter. Each datacenter can run multiple clusters. This configuration reduces the risk of a hardware failure or network outage that can cause your cluster to fail. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

The default installation configuration deploys a cluster to a single vSphere datacenter. If you want to deploy a cluster to multiple vSphere datacenters, you must create an installation configuration file that enables the region and zone feature.

The default `install-config.yaml` file includes `vccenters` and `failureDomains` fields, where you can specify

multiple vSphere datacenters and clusters for your OpenShift Container Platform cluster. You can leave these fields blank if you want to install an OpenShift Container Platform cluster in a vSphere environment that consists of single datacenter.

The following list describes terms associated with defining zones and regions for your cluster:

- **Failure domain:** Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a **datastore** object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes.
- **Region:** Specifies a vCenter datacenter. You define a region by using a tag from the **openshift-region** tag category.
- **Zone:** Specifies a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category.



NOTE

If you plan on specifying more than one failure domain in your **install-config.yaml** file, you must create tag categories, zone tags, and region tags in advance of creating the configuration file.

You must create a vCenter tag for each vCenter datacenter, which represents a region. Additionally, you must create a vCenter tag for each cluster that runs in a datacenter, which represents a zone. After you create the tags, you must attach each tag to their respective datacenters and clusters.

The following table outlines an example of the relationship among regions, zones, and tags for a configuration with multiple vSphere datacenters running in a single VMware vCenter.

Table 24.97. Example of a configuration with multiple vSphere datacenters that run in a single VMware vCenter

Datacenter (region)	Cluster (zone)	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

24.8.12. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

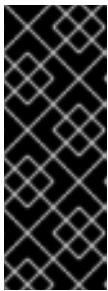
Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- Obtain the **imageContentSources** section from the output of the command to mirror the repository.
- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

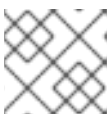
```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

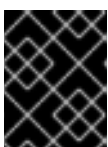
2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

- Unless you use a registry that RHCOS trusts by default, such as **docker.io**, you must provide the contents of the certificate for your mirror repository in the **additionalTrustBundle** section. In most cases, you must provide the certificate for your mirror.
 - You must include the **imageContentSources** section from the output of the command to mirror the repository.
3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

24.8.12.1. Sample install-config.yaml file for VMware vSphere

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
  name: worker
  replicas: 0 3
controlPlane: 4
  name: master
  replicas: 3 5
metadata:
  name: test 6
platform:
  vsphere:
    vcenter: your.vcenter.server 7
    username: username 8
    password: password 9
    datacenter: datacenter 10
    defaultDatastore: datastore 11
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 12
    resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 13
    diskType: thin 14
  fips: false 15
pullSecret: '{"auths":{"<local_registry>":{"auth": "<credentials>","email": "you@example.com"}}}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
additionalTrustBundle: | 18
  -----BEGIN CERTIFICATE-----
  /-----END CERTIFICATE-----
imageContentSources: 19
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release
  source: <source_image_1>
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release-images
  source: <source_image_2>

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 4 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, (-), and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
- 3 You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

- 5 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control
- 6 The cluster name that you specified in your DNS records.
- 7 The fully-qualified hostname or IP address of the vCenter server.



IMPORTANT

The Cluster Cloud Controller Manager Operator performs a connectivity check on a provided hostname or IP address. Ensure that you specify a hostname or an IP address to a reachable vCenter server. If you provide metadata to a non-existent vCenter server, installation of the cluster fails at the bootstrap stage.

- 8 The name of the user for accessing the server.
- 9 The password associated with the vSphere user.
- 10 The vSphere datacenter.
- 11 The default vSphere datastore to use.
- 12 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster and you do not want to use the default **StorageClass** object, named **thin**, you can omit the **folder** parameter from the **install-config.yaml** file.
- 13 Optional parameter: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.
- 14 The vSphere disk provisioning method.
- 15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 16 For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For `<credentials>`, specify the base64-encoded user name and

`registry.example.com:5000`. For `credentials`, specify the base64-encoded user name and password for your mirror registry.

- 17 The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

- 18 Provide the contents of the certificate file that you used for your mirror registry.
- 19 Provide the **imageContentSources** section from the output of the command to mirror the repository.

24.8.12.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

- Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
```

```
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
```

```
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

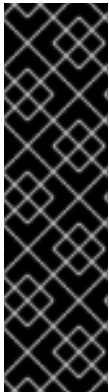


NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

24.8.12.3. Configuring regions and zones for a VMware vCenter

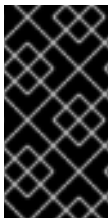
You can modify the default installation configuration file to deploy an OpenShift Container Platform cluster to multiple vSphere datacenters that run in a single VMware vCenter.



IMPORTANT

VMware vSphere region and zone enablement is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

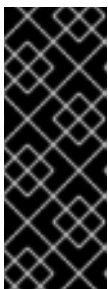


IMPORTANT

The example uses the **govc** command. The **govc** command is an open source command available from VMware. The **govc** command is not available from Red Hat. Red Hat Support does not maintain the **govc** command. Instructions for downloading and installing **govc** are found on the VMware documentation website.

Prerequisites

- You have an existing **install-config.yaml** installation configuration file.



IMPORTANT

You must specify at least one failure domain for your OpenShift Container Platform cluster, so that you can provision datacenter objects for your VMware vCenter server. Consider specifying multiple failure domains if you need to provision virtual machine nodes in different datacenters, clusters, datastores, and other components. To enable regions and zones, you must define multiple failure domains for your OpenShift Container Platform cluster.



NOTE

You cannot change a failure domain after you installed an OpenShift Container Platform cluster on the VMware vSphere platform. You can add additional failure domains after cluster installation.

Procedure

- Enter the following **govc** command-line tool commands to create the **openshift-region** and **openshift-zone** vCenter tag categories:



IMPORTANT

If you specify different names for the **openshift-region** and **openshift-zone** vCenter tag categories, the installation of the OpenShift Container Platform cluster fails.

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

- To create a region tag for each region vSphere datacenter where you want to deploy your cluster, enter the following command in your terminal:

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

- To create a zone tag for each vSphere cluster where you want to deploy your cluster, enter the following command:

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

- Attach region tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

- Attach the zone tags to each vCenter datacenter object by entering the following command:

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

- Change to the directory that contains the installation program and initialize the cluster deployment according to your chosen installation requirements.

Sample install-config.yaml file with multiple datacenters defined in a vSphere center

```
apiVersion: v1
baseDomain: example.com
featureSet: TechPreviewNoUpgrade 1
compute:
  name: worker
  replicas: 3
  vsphere:
    zones: 2
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
controlPlane:
  name: master
  replicas: 3
  vsphere:
    zones: 3
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
metadata:
  name: cluster
platform:
  vsphere:
    vcenter: <vcenter_server> 4
    username: <username> 5
    password: <password> 6
    datacenter: datacenter 7
    defaultDatastore: datastore 8
```

```

folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 9
cluster: cluster 10
resourcePool: "/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>" 11
diskType: thin
failureDomains: 12
- name: <machine_pool_zone_1> 13
  region: <region_tag_1> 14
  zone: <zone_tag_1> 15
  topology: 16
    datacenter: <datacenter1> 17
    computeCluster: "/<datacenter1>/host/<cluster1>" 18
    resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>" 19
    networks: 20
      - <VM_Network1_name>
    datastore: "/<datacenter1>/datastore/<datastore1>" 21
- name: <machine_pool_zone_2>
  region: <region_tag_2>
  zone: <zone_tag_2>
  topology:
    datacenter: <datacenter2>
    computeCluster: "/<datacenter2>/host/<cluster2>"
    networks:
      - <VM_Network2_name>
    datastore: "/<datacenter2>/datastore/<datastore2>"
    resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
    folder: "/<datacenter2>/vm/<folder2>"
# ...

```

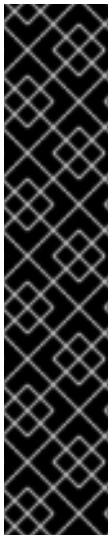
- 1 You must define set the **TechPreviewNoUpgrade** as the value for this parameter, so that you can use the VMware vSphere region and zone enablement feature.
- 2 3 An optional parameter for specifying a vCenter cluster. You define a zone by using a tag from the **openshift-zone** tag category. If you do not define this parameter, nodes will be distributed among all defined failure-domains.
- 4 5 6 7 8 9 10 11 The default vCenter topology. The installation program uses this topology information to deploy the bootstrap node. Additionally, the topology defines the default datastore for vSphere persistent volumes.
- 12 Establishes the relationships between a region and zone. You define a failure domain by using vCenter objects, such as a datastore object. A failure domain defines the vCenter location for OpenShift Container Platform cluster nodes. If you do not define this parameter, the installation program uses the default vCenter topology.
- 13 Defines the name of the failure domain. Each failure domain is referenced in the **zones** parameter to scope a machine pool to the failure domain.
- 14 You define a region by using a tag from the **openshift-region** tag category. The tag must be attached to the vCenter datacenter.
- 15 You define a zone by using a tag from the **openshift-zone tag** category. The tag must be attached to the vCenter datacenter.
- 16 Specifies the vCenter resources associated with the failure domain.

- 17 An optional parameter for defining the vSphere datacenter that is associated with a failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 18 An optional parameter for stating the absolute file path for the compute cluster that is associated with the failure domain. If you do not define this parameter, the installation program uses the default vCenter topology.
- 19 An optional parameter for the installer-provisioned infrastructure. The parameter sets the absolute path of an existing resource pool where the installation program creates the virtual machines, for example, `/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>`. If you do not specify a value, resources are installed in the root of the cluster `/example_datacenter/host/example_cluster/Resources`.
- 20 An optional parameter that lists any network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured. If you do not define this parameter, the installation program uses the default vCenter topology.
- 21 An optional parameter for specifying a datastore to use for provisioning volumes. If you do not define this parameter, the installation program uses the default vCenter topology.

24.8.13. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the compute machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.
3. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
 4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

24.8.14. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the **jq** package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6 1
```

- 1 The output of this command is your cluster name and a random string.

24.8.15. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on user-provisioned infrastructure on VMware vSphere, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on vSphere hosts. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

Procedure

1. Upload the bootstrap Ignition config file, which is named **<installation_directory>/bootstrap.ign**, that the installation program created to your HTTP server. Note the URL of this file.
2. Save the following secondary Ignition config file for your bootstrap node to your computer as **<installation_directory>/merge-bootstrap.ign**:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

- 1** Specify the URL of the bootstrap Ignition config file that you hosted.

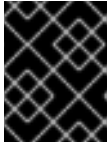
When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:
 - **<installation_directory>/master.ign**
 - **<installation_directory>/worker.ign**
 - **<installation_directory>/merge-bootstrap.ign**
4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter **guestinfo.ignition.config.data** in your VM. For example, if you use a Linux operating system, you can use the **base64** command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the [RHCOS image mirror](#) page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format **rhcos-vmware.<architecture>.ova**.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
 - a. Click the **VMs and Templates** view.
 - b. Right-click the name of your datacenter.
 - c. Click **New Folder → New VM and Template Folder**.
 - d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

- a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
- b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
- c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
- d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
- e. On the **Select storage** tab, configure the storage options for your VM.
 - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.

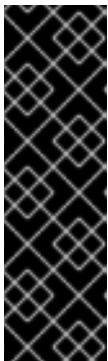
- Select the datastore that you specified in your **install-config.yaml** file.
- f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
 - g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.



IMPORTANT

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that compute machine sets can apply configurations to.

8. Optional: Update the configured virtual hardware version in the VM template, if necessary. Follow [Upgrading a virtual machine to the latest hardware version](#) in the VMware documentation for more information.



IMPORTANT

It is recommended that you update the hardware version of the VM template to version 15 before creating VMs from it, if necessary. Using hardware version 13 for your cluster nodes running on vSphere is now deprecated. If your imported template defaults to hardware version 13, you must ensure that your ESXi host is on 6.7U3 or later before upgrading the VM template to hardware version 15. If your vSphere version is less than 6.7U3, you can skip this upgrade step; however, a future version of OpenShift Container Platform is scheduled to remove support for hardware version 13 and vSphere versions less than 6.7U3.

9. After the template deploys, deploy a VM for a machine in the cluster.
 - a. Right-click the template name and click **Clone** → **Clone to Virtual Machine**
 - b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

- c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
- d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
- e. On the **Select clone options** tab, select **Customize this virtual machine's hardware**
- f. On the **Customize hardware** tab, click **Advanced Parameters**.



IMPORTANT

The following configuration suggestions are for example purposes only. As a cluster administrator, you must configure resources according to the resource demands placed on your cluster. To best manage cluster resources, consider creating a resource pool from the cluster's root resource pool.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
 - Set your static IP configuration:

Example command

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:<iface>:none
nameserver=svr1 [nameserver=svr2 [nameserver=svr3 [...]]]"
```

Example command

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
nameserver=8.8.8.8"
```

- Set the **guestinfo.afterburn.initrd.network-kargs** property before you boot a VM from an OVA in vSphere:

Example command

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-
kargs=${IPCFG}"
```

- Add the following configuration parameter names and values by specifying data in the **Attribute** and **Values** fields. Ensure that you select the **Add** button for each parameter that you create.
 - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
 - **stealclock.enable**: If this parameter was not defined, add it and specify **TRUE**.
 - Create a child resource pool from the cluster's root resource pool. Perform resource allocation in this child resource pool.
- g. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- h. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
- i. From the **Virtual Machines** tab, right-click on your VM and then select **Power** → **Power On**.

- j. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

Next steps

- Create the rest of the machines for your cluster by following the preceding steps for each machine.



IMPORTANT

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

24.8.16. Adding more compute machines to a cluster in vSphere

You can add more compute machines to a user-provisioned OpenShift Container Platform cluster on VMware vSphere.

After your vSphere template deploys in your OpenShift Container Platform cluster, you can deploy a virtual machine (VM) for a machine in that cluster.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. Right-click the template's name and click **Clone → Clone to Virtual Machine**
2. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.



NOTE

Ensure that all virtual machine names across a vSphere installation are unique.

3. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
4. On the **Select a compute resource** tab, select the name of a host in your datacenter.
5. On the **Select storage** tab, select storage for your configuration and disk files.
6. On the **Select clone options**, select **Customize this virtual machine's hardware**
7. On the **Customize hardware** tab, click **Advanced**.

- a. Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
 - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
 - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
 - **disk.EnableUUID**: Specify **TRUE**.
8. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. If many networks exist, select **Add New Device > Network Adapter**, and then enter your network information in the fields provided by the **New Network** menu item.
9. Complete the remaining configuration steps. On clicking the **Finish** button, you have completed the cloning operation.
10. From the **Virtual Machines** tab, right-click on your VM and then select **Power → Power On**.

Next steps

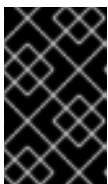
- Continue to create more compute machines for your cluster.

24.8.17. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

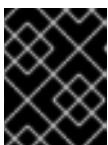
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions**: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate **/var** partition. See "Creating a separate **/var** partition" and this [Red Hat Knowledgebase article](#) for more information.



IMPORTANT

Kubernetes supports only two file system partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- **Retain existing partitions**: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to **coreos-installer** that allow you to retain existing data partitions.

Creating a separate **/var** partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
```



```

metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> ❶
      partitions:
        - label: var
          start_mib: <partition_start_offset> ❷
          size_mib: <partition_size> ❸
          number: 5
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          mount_options: [defaults, prjquota] ❹
          with_mount_unit: true

```

- ❶ The storage device name of the disk that you want to partition.
- ❷ When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
- ❸ The size of the data partition in mebibytes.
- ❹ The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

24.8.18. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1  
--log-level=info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...  
INFO API v1.25.0 up  
INFO Waiting up to 30m0s for bootstrapping to complete...  
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

24.8.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container

Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

24.8.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES    AGE    VERSION
master-0  Ready    master   63m    v1.25.0
master-1  Ready    master   63m    v1.25.0
master-2  Ready    master   64m    v1.25.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-8b2br 15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m   system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

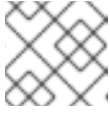
- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



NOTE

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE    REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
```

```

master-2 Ready   master 74m v1.25.0
worker-0 Ready   worker 11m v1.25.0
worker-1 Ready   worker 11m v1.25.0

```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

24.8.21. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m

node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

24.8.21.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

24.8.21.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

24.8.21.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.

- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.



IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

3. Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
■
```



```
storage:
  pvc:
    claim: 1
```

- 1 Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** persistent volume claim (PVC). The PVC is generated based on the default storage class. However, be aware that the default storage class might provide ReadWriteOnce (RWO) volumes, such as a RADOS Block Device (RBD), which can cause issues when you replicate to more than one replica.

4. Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
image-registry	4.7	True	False	6h50m

24.8.21.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

24.8.21.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
    - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1** A unique name that represents the **PersistentVolumeClaim** object.
- 2** The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3** The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4** The size of the persistent volume claim.

- b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
pvc:
claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring registry storage for VMware vSphere](#).

24.8.22. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m

node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE          NAME                                     READY STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
```

```

Running 1 9m
openshift-apiserver apiserver-67b9g 1/1 Running 0
3m
openshift-apiserver apiserver-ljcmx 1/1 Running 0
1m
openshift-apiserver apiserver-z25h4 1/1 Running 0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8 1/1
Running 0 5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1 Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

- For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.
- Register your cluster on the [Cluster registration](#) page.

You can add extra compute machines after the cluster installation is completed by following [Adding compute machines to vSphere](#).

24.8.23. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

Procedure

To create a backup of persistent volumes:

- Stop the application that is using the persistent volume.
- Clone the persistent volume.
- Restart the application.
- Create a backup of the cloned volume.
- Delete the cloned volume.

24.8.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

24.8.25. Next steps

- [Customize your cluster](#).
- [Configure image streams](#) for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to [use Operator Lifecycle Manager \(OLM\) on restricted networks](#).
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by [configuring additional trust stores](#).
- If necessary, you can [opt out of remote health reporting](#).
- Optional: [View the events from the vSphere Problem Detector Operator](#) to determine if the cluster has permission or storage configuration issues.

24.9. UNINSTALLING A CLUSTER ON VMC

You can remove a cluster installed on VMware vSphere infrastructure that you deployed to [VMware Cloud \(VMC\) on AWS](#) by using installer-provisioned infrastructure.

24.9.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with user-provisioned infrastructure clusters. There might be resources that the installation program did not create or that the installation program is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. On the computer that you used to install the cluster, go to the directory that contains the installation program, and run the following command:

```
$. /openshift-install destroy cluster \  
--dir <installation_directory> --log-level info 1 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.
- 2 To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 25. INSTALLING ON ANY PLATFORM

25.1. INSTALLING A CLUSTER ON ANY PLATFORM

In OpenShift Container Platform version 4.12, you can install a cluster on any infrastructure that you provision, including virtualization and cloud environments.



IMPORTANT

Review the information in the [guidelines for deploying OpenShift Container Platform on non-tested platforms](#) before you attempt to install an OpenShift Container Platform cluster in virtualized or cloud environments.

25.1.1. Prerequisites

- You reviewed details about the [OpenShift Container Platform installation and update](#) processes.
- You read the documentation on [selecting a cluster installation method and preparing it for users](#).
- If you use a firewall, you [configured it to allow the sites](#) that your cluster requires access to.



NOTE

Be sure to also review this site list if you are configuring a proxy.

25.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.12, you require access to the internet to install your cluster.

You must have internet access to:

- Access [OpenShift Cluster Manager Hybrid Cloud Console](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.



IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

25.1.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

25.1.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 25.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.



IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

25.1.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 25.2. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.
2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.
3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

25.1.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

25.1.3.4. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

During the initial boot, the machines require an IP address configuration that is set either through a DHCP server or statically by providing the required boot options. After a network connection is established, the machines download their Ignition config files from an HTTP or HTTPS server. The Ignition config files are then used to set the exact state of each machine. The Machine Config Operator completes more changes to the machines, such as the application of new certificates or keys, after installation.

It is recommended to use a DHCP server for long-term management of the cluster machines. Ensure that the DHCP server is configured to provide persistent IP addresses, DNS server information, and hostnames to the cluster machines.

**NOTE**

If a DHCP service is not available for your user-provisioned infrastructure, you can instead provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

25.1.3.4.1. Setting the cluster node hostnames through DHCP

On Red Hat Enterprise Linux CoreOS (RHCOS) machines, the hostname is set through NetworkManager. By default, the machines obtain their hostname through DHCP. If the hostname is not provided by DHCP, set statically through kernel arguments, or another method, it is obtained through a reverse DNS lookup. Reverse DNS lookup occurs after the network has been initialized on a node and can take time to resolve. Other system services can start prior to this and detect the hostname as **localhost** or similar. You can avoid this by using DHCP to provide the hostname for each cluster node.

Additionally, setting the hostnames through DHCP can bypass any manual DNS record name configuration errors in environments that have a DNS split-horizon implementation.

25.1.3.4.2. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

**IMPORTANT**

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 25.3. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	1936	Metrics
	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099 .
	10250-10259	The default ports that Kubernetes reserves

Protocol	Port	Description
	10256	openshift-sdn
UDP	4789	VXLAN
	6081	Geneve
	9000-9999	Host level services, including the node exporter on ports 9100-9101 .
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port 123 If an external NTP time server is configured, you must open UDP port 123 .
TCP/UDP	30000-32767	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 25.4. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	6443	Kubernetes API

Table 25.5. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	2379-2380	etcd server and peer ports

NTP configuration for user-provisioned infrastructure

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- [Configuring chrony time service](#)

25.1.3.5. User-provisioned DNS requirements

In OpenShift Container Platform deployments, DNS name resolution is required for the following components:

- The Kubernetes API
- The OpenShift Container Platform application wildcard
- The bootstrap, control plane, and compute machines

Reverse DNS resolution is also required for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.

DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the hostnames for all the nodes, unless the hostnames are provided by DHCP. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.




NOTE

It is recommended to use a DHCP server to provide the hostnames to each cluster node. See the *DHCP recommendations for user-provisioned infrastructure* section for more information.

The following DNS records are required for a user-provisioned OpenShift Container Platform cluster and they must be in place before installation. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the base domain that you specify in the **install-config.yaml** file. A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>..**

Table 25.6. Required DNS records

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the API load balancer. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
	api-int.<cluster_name>.<base_domain>..	A DNS A/AAAA or CNAME record, and a DNS PTR record, to internally identify the API load balancer. These records must be resolvable from all the nodes within the cluster.
		 <p>IMPORTANT</p> <p>The API server must be able to resolve the worker nodes by the hostnames that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</p>

Component	Record	Description
Routes	*.apps.<cluster_name>.<base_domain>.	<p>A wildcard DNS A/AAAA or CNAME record that refers to the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</p> <p>For example, console-openshift-console.apps.<cluster_name>.<base_domain> is used as a wildcard route to the OpenShift Container Platform console.</p>
Bootstrap machine	bootstrap.<cluster_name>.<base_domain>.	A DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.
Control plane machines	<control_plane><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes. These records must be resolvable by the nodes within the cluster.
Compute machines	<compute><n>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.



NOTE

In OpenShift Container Platform 4.4 and later, you do not need to specify etcd host and SRV records in your DNS configuration.

TIP

You can use the **dig** command to verify name and reverse name resolution. See the section on *Validating DNS resolution for user-provisioned infrastructure* for detailed validation steps.

25.1.3.5.1. Example DNS configuration for user-provisioned clusters

This section provides A and PTR record configuration samples that meet the DNS requirements for deploying OpenShift Container Platform on user-provisioned infrastructure. The samples are not meant to provide advice for choosing one DNS solution over another.

In the examples, the cluster name is **ocp4** and the base domain is **example.com**.

Example DNS A record configuration for a user-provisioned cluster

The following example is a BIND zone file that shows sample A records for name resolution in a user-provisioned cluster.

Example 25.1. Sample DNS zone database

```

$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
;
;
ns1.example.com. IN A 192.168.1.5
smtp.example.com. IN A 192.168.1.5
;
helper.example.com. IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com. IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF

```

- 1 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer.
- 2 Provides name resolution for the Kubernetes API. The record refers to the IP address of the API load balancer and is used for internal cluster communications.
- 3 Provides name resolution for the wildcard routes. The record refers to the IP address of the application ingress load balancer. The application ingress load balancer targets the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

**NOTE**

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

- 4 Provides name resolution for the bootstrap machine.
- 5 6 7 Provides name resolution for the control plane machines.
- 8 9 Provides name resolution for the compute machines.

Example DNS PTR record configuration for a user-provisioned cluster

The following example BIND zone file shows sample PTR records for reverse name resolution in a user-provisioned cluster.

Example 25.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H ; refresh (3 hours)
    30M ; retry (30 minutes)
    2W ; expiry (2 weeks)
    1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

- 1 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer.
- 2 Provides reverse DNS resolution for the Kubernetes API. The PTR record refers to the record name of the API load balancer and is used for internal cluster communications.
- 3 Provides reverse DNS resolution for the bootstrap machine.
- 4 5 6 Provides reverse DNS resolution for the control plane machines.
- 7 8 Provides reverse DNS resolution for the compute machines.

**NOTE**

A PTR record is not required for the OpenShift Container Platform application wildcard.

25.1.3.6. Load balancing requirements for user-provisioned infrastructure

Before you install OpenShift Container Platform, you must provision the API and application ingress load balancing infrastructure. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

**NOTE**

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
 - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
 - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 25.7. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer.** Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.

Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

TIP

If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

Configure the following ports on both the front and back of the load balancers:

Table 25.8. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic

**NOTE**

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

25.1.3.6.1. Example load balancer configuration for user-provisioned clusters

This section provides an example API and application ingress load balancer configuration that meets the load balancing requirements for user-provisioned clusters. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.



NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 25.3. Sample API and application Ingress load balancer configuration

```

global
  log      127.0.0.1 local2
  pidfile  /var/run/haproxy.pid
  maxconn  4000
  daemon
defaults
  mode            http
  log             global
  option          dontlognull
  option http-server-close
  option          redispatch
  retries         3
  timeout http-request 10s
  timeout queue    1m
  timeout connect  10s
  timeout client   1m
  timeout server   1m
  timeout http-keep-alive 10s
  timeout check    10s
  maxconn         3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option httpchk GET /readyz HTTP/1.0
  option log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
  rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
  fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5

```

```

bind *:443
mode tcp
balance source
server worker0 worker0.ocp4.example.com:443 check inter 1s
server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
bind *:80
mode tcp
balance source
server worker0 worker0.ocp4.example.com:80 check inter 1s
server worker1 worker1.ocp4.example.com:80 check inter 1s

```

- 1** Port **6443** handles the Kubernetes API traffic and points to the control plane machines.
- 2** **4** The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.
- 3** Port **22623** handles the machine config server traffic and points to the control plane machines.
- 5** Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.
- 6** Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltp** on the HAProxy node.

25.1.4. Preparing the user-provisioned infrastructure

Before you install OpenShift Container Platform on user-provisioned infrastructure, you must prepare the underlying infrastructure.

This section provides details about the high-level steps required to set up your cluster infrastructure in preparation for an OpenShift Container Platform installation. This includes configuring IP networking and network connectivity for your cluster nodes, enabling the required ports through your firewall, and setting up the required DNS and load balancing infrastructure.

After preparation, your cluster infrastructure must meet the requirements outlined in the *Requirements for a cluster with user-provisioned infrastructure* section.

Prerequisites

- You have reviewed the [OpenShift Container Platform 4.x Tested Integrations](#) page.
- You have reviewed the infrastructure requirements detailed in the *Requirements for a cluster with user-provisioned infrastructure* section.

Procedure

1. If you are using DHCP to provide the IP networking configuration to your cluster nodes, configure your DHCP service.
 - a. Add persistent IP addresses for the nodes to your DHCP server configuration. In your configuration, match the MAC address of the relevant network interface to the intended IP address for each node.
 - b. When you use DHCP to configure IP addressing for the cluster machines, the machines also obtain the DNS server information through DHCP. Define the persistent DNS server address that is used by the cluster nodes through your DHCP server configuration.



NOTE

If you are not using a DHCP service, you must provide the IP networking configuration and the address of the DNS server to the nodes at RHCOS install time. These can be passed as boot arguments if you are installing from an ISO image. See the *Installing RHCOS and starting the OpenShift Container Platform bootstrap process* section for more information about static IP provisioning and advanced networking options.

- c. Define the hostnames of your cluster nodes in your DHCP server configuration. See the *Setting the cluster node hostnames through DHCP* section for details about hostname considerations.



NOTE

If you are not using a DHCP service, the cluster nodes obtain their hostname through a reverse DNS lookup.

2. Ensure that your network infrastructure provides the required network connectivity between the cluster components. See the *Networking requirements for user-provisioned infrastructure* section for details about the requirements.
3. Configure your firewall to enable the ports required for the OpenShift Container Platform cluster components to communicate. See *Networking requirements for user-provisioned infrastructure* section for details about the ports that are required.



IMPORTANT

By default, port **1936** is accessible for an OpenShift Container Platform cluster, because each control plane node needs access to this port.

Avoid using the Ingress load balancer to expose this port, because doing so might result in the exposure of sensitive information, such as statistics and metrics, related to Ingress Controllers.

4. Setup the required DNS infrastructure for your cluster.

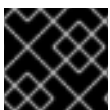
- a. Configure DNS name resolution for the Kubernetes API, the application wildcard, the bootstrap machine, the control plane machines, and the compute machines.
 - b. Configure reverse DNS resolution for the Kubernetes API, the bootstrap machine, the control plane machines, and the compute machines.
See the *User-provisioned DNS requirements* section for more information about the OpenShift Container Platform DNS requirements.
5. Validate your DNS configuration.
- a. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses in the responses correspond to the correct components.
 - b. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names in the responses correspond to the correct components.
See the *Validating DNS resolution for user-provisioned infrastructure* section for detailed DNS validation steps.
6. Provision the required API and application ingress load balancing infrastructure. See the *Load balancing requirements for user-provisioned infrastructure* section for more information about the requirements.

**NOTE**

Some load balancing solutions require the DNS name resolution for the cluster nodes to be in place before the load balancing is initialized.

25.1.5. Validating DNS resolution for user-provisioned infrastructure

You can validate your DNS configuration before installing OpenShift Container Platform on user-provisioned infrastructure.

**IMPORTANT**

The validation steps detailed in this section must succeed before you install your cluster.

Prerequisites

- You have configured the required DNS records for your user-provisioned infrastructure.

Procedure

1. From your installation node, run DNS lookups against the record names of the Kubernetes API, the wildcard routes, and the cluster nodes. Validate that the IP addresses contained in the responses correspond to the correct components.
 - a. Perform a lookup against the Kubernetes API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

- 1** Replace **<nameserver_ip>** with the IP address of the nameserver, **<cluster_name>** with your cluster name, and **<base_domain>** with your base domain name.

Example output

```
api.ocp4.example.com. 604800 IN A 192.168.1.5
```

- b. Perform a lookup against the Kubernetes internal API record name. Check that the result points to the IP address of the API load balancer:

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

Example output

```
api-int.ocp4.example.com. 604800 IN A 192.168.1.5
```

- c. Test an example ***.apps.<cluster_name>.<base_domain>** DNS wildcard lookup. All of the application wildcard lookups must resolve to the IP address of the application ingress load balancer:

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

Example output

```
random.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```



NOTE

In the example outputs, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

You can replace **random** with another wildcard value. For example, you can query the route to the OpenShift Container Platform console:

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.<cluster_name>.<base_domain>
```

Example output

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

- d. Run a lookup against the bootstrap DNS record name. Check that the result points to the IP address of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

Example output

```
bootstrap.ocp4.example.com. 604800 IN A 192.168.1.96
```

- e. Use this method to perform lookups against the DNS record names for the control plane and compute nodes. Check that the results correspond to the IP addresses of each node.
2. From your installation node, run reverse DNS lookups against the IP addresses of the load balancer and the cluster nodes. Validate that the record names contained in the responses correspond to the correct components.
 - a. Perform a reverse lookup against the IP address of the API load balancer. Check that the response includes the record names for the Kubernetes API and the Kubernetes internal API:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

Example output

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 Provides the record name for the Kubernetes internal API.

2 Provides the record name for the Kubernetes API.



NOTE

A PTR record is not required for the OpenShift Container Platform application wildcard. No validation step is needed for reverse DNS resolution against the IP address of the application ingress load balancer.

- b. Perform a reverse lookup against the IP address of the bootstrap node. Check that the result points to the DNS record name of the bootstrap node:

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

Example output

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

- c. Use this method to perform reverse lookups against the IP addresses for the control plane and compute nodes. Check that the results correspond to the DNS record names of each node.

25.1.6. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the `~/.ssh/authorized_keys` list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The `./openshift-install gather` command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.



NOTE

You must use a local key, not one that you configured with platform-specific approaches such as [AWS key pairs](#).

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
```

- 1 Specify the path and file name, such as `~/.ssh/id_ed25519`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the `x86_64`, `ppc64le`, and `s390x` architectures, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the `~/.ssh/id_ed25519.pub` public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the `./openshift-install gather` command.



NOTE

On some distributions, default SSH private key identities such as `~/.ssh/id_rsa` and `~/.ssh/id_dsa` are managed automatically.

- a. If the `ssh-agent` process is not already running for your local user, start it as a background task:

■

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

- 1** Specify the path and file name for your SSH private key, such as `~/.ssh/id_ed25519`

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

25.1.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Access the [Infrastructure Provider](#) page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program that corresponds with your host operating system and architecture, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. Download your installation [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

25.1.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.12. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.12 Linux Client** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.

To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the [OpenShift Container Platform downloads page](#) on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.12 macOS Client** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.12 macOS arm64 Client** entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

25.1.9. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.



IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

25.1.9.1. Sample install-config.yaml file for other platforms

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

```

apiVersion: v1
baseDomain: example.com 1
compute: 2
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16

```

- 1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
- 2 5 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Only one control plane pool is used.
- 3 6 Specifies whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of the cores in your machines. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.



NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.



IMPORTANT

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml** file, ensure that your capacity planning accounts for the dramatically decreased machine performance.

- 4 You must set this value to **0** when you install OpenShift Container Platform on user-provisioned infrastructure. In installer-provisioned installations, the parameter controls the number of compute



NOTE

If you are installing a three-node cluster, do not deploy any compute machines when you install the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

- 7 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.
- 8 The cluster name that you specified in your DNS records.
- 9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.



NOTE

Class E CIDR range is reserved for a future use. To use the Class E CIDR range, you must ensure your networking environment accepts the IP addresses within the Class E CIDR range.

- 10 The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23**, then each node is assigned a **/23** subnet out of the given **cidr**, which allows for 510 ($2^{(32 - 23)} - 2$) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.
- 11 The cluster network plugin to install. The supported values are **OVNKubernetes** and **OpenShiftSDN**. The default value is **OVNKubernetes**.
- 12 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.
- 13 You must set the platform to **none**. You cannot provide additional platform configuration variables for your platform.



IMPORTANT

Clusters that are installed with the platform type **none** are unable to use some features, such as managing compute machines with the Machine API. This limitation applies even if the compute machines that are attached to the cluster are installed on a platform that would normally support the feature. This parameter cannot be changed after installation.

- 14 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#). The use of FIPS validated or Modules In Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64**, **ppc64le**, and **s390x** architectures.

- 15 The [pull secret from the Red Hat OpenShift Cluster Manager](#). This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
- 16 The SSH public key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

25.1.9.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.



NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
```



```

proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5

```

- 1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
- 2 A proxy URL to use for creating HTTPS connections outside the cluster.
- 3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.
- 4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
- 5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.



NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

25.1.9.3. Configuring a three-node cluster

Optionally, you can deploy zero compute machines in a bare metal cluster that consists of three control plane machines only. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for testing, development, and production.

In three-node OpenShift Container Platform environments, the three control plane machines are schedulable, which means that your application workloads are scheduled to run on them.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Ensure that the number of compute replicas is set to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

**NOTE**

You must set the value of the **replicas** parameter for the compute machines to **0** when you install OpenShift Container Platform on user-provisioned infrastructure, regardless of the number of compute machines you are deploying. In installer-provisioned installations, the parameter controls the number of compute machines that the cluster creates and manages for you. This does not apply to user-provisioned installations, where the compute machines are deployed manually.

For three-node cluster installations, follow these next steps:

- If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes. See the *Load balancing requirements for user-provisioned infrastructure* section for more information.
- When you create the Kubernetes manifest files in the following procedure, ensure that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file is set to **true**. This enables your application workloads to run on the control plane nodes.
- Do not deploy any compute nodes when you create the Red Hat Enterprise Linux CoreOS (RHCOS) machines.

25.1.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

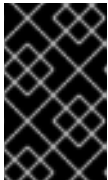
```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1** For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.



WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.



IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

2. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:
 - a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.
 - b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.
 - c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

25.1.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.

**NOTE**

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Only RHEL 8 compute machines are supported.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments:** You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the **APPEND** parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special **coreos.inst.*** arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.
- **Ignition configs:** OpenShift Container Platform Ignition config files (***.ign**) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the **coreos-installer** to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.
- **coreos-installer:** You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the **coreos-installer** command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.

**NOTE**

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

25.1.11.1. Installing RHCOS by using an ISO image

You can use an ISO image to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.

- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

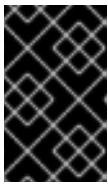
Procedure

- Obtain the SHA512 digest for each of your Ignition config files. For example, you can use the following on a system running Linux to get the SHA512 digest for your **bootstrap.ign** Ignition config file:

```
$ sha512sum <installation_directory>/bootstrap.ign
```

The digests are provided to the **coreos-installer** in a later step to validate the authenticity of the Ignition config files on the cluster nodes.

- Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

- From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left    Speed
  0   0   0   0   0   0   0   0   0  --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

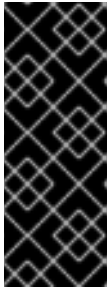
- Although it is possible to obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS images are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep '\.iso[^\.]'
```

Example output

```
"location": "<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-
```

```
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

5. Use the ISO to start the RHCOS installation. Use one of the following installation options:
 - Burn the ISO image to a disk and boot it directly.
 - Use ISO redirection by using a lights-out management (LOM) interface.
6. Boot the RHCOS ISO image without specifying any options or interrupting the live boot sequence. Wait for the installer to boot into a shell prompt in the RHCOS live environment.



NOTE

It is possible to interrupt the RHCOS installation boot process to add kernel arguments. However, for this ISO procedure you should use the **coreos-installer** command as outlined in the following steps, instead of adding kernel arguments.

7. Run the **coreos-installer** command and specify the options that meet your installation requirements. At a minimum, you must specify the URL that points to the Ignition config file for the node type, and the device that you are installing to:

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

- 1 1 You must run the **coreos-installer** command by using **sudo**, because the **core** user does not have the required root privileges to perform the installation.
- 2 The **--ignition-hash** option is required when the Ignition config file is obtained through an HTTP URL to validate the authenticity of the Ignition config file on the cluster node. **<digest>** is the Ignition config file SHA512 digest obtained in a preceding step.



NOTE

If you want to provide your Ignition config files through an HTTPS server that uses TLS, you can add the internal certificate authority (CA) to the system trust store before running **coreos-installer**.

The following example initializes a bootstrap node installation to the **/dev/sda** device. The Ignition config file for the bootstrap node is obtained from an HTTP web server with the IP address 192.168.1.2:

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

9. After RHCOS installs, you must reboot the system. During the system reboot, it applies the Ignition config file that you specified.
10. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. Continue to create the other machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install OpenShift Container Platform.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

25.1.11.2. Installing RHCOS by using PXE or iPXE booting

You can use PXE or iPXE booting to install RHCOS on the machines.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have configured suitable PXE or iPXE infrastructure.
- You have an HTTP server that can be accessed from your computer, and from the machines that you create.
- You have reviewed the *Advanced RHCOS installation configuration* section for different ways to configure features, such as networking and disk partitioning.

Procedure

1. Upload the bootstrap, control plane, and compute node Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.



IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. From the installation host, validate that the Ignition config files are available on the URLs. The following example gets the Ignition config file for the bootstrap node:

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

Example output

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total   Spent    Left    Speed
  0   0   0    0    0    0    0    0  0 --:--:-- --:--:-- --:--:--    0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

Replace **bootstrap.ign** with **master.ign** or **worker.ign** in the command to validate that the Ignition config files for the control plane and compute nodes are also available.

3. Although it is possible to obtain the RHCOS **kernel**, **initramfs** and **rootfs** files that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page, the recommended way to obtain the correct version of your RHCOS files are from the output of **openshift-install** command:

```
$ openshift-install coreos print-stream-json | grep -Eo "https.*(kernel-|initramfs.|rootfs.)w+(\.img)?"
```

Example output

■

```
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.12/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS QCOW2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel:** rhcos-<version>-live-kernel-<architecture>
- **initramfs:** rhcos-<version>-live-initramfs.<architecture>.img
- **rootfs:** rhcos-<version>-live-rootfs.<architecture>.img

4. Upload the **rootfs**, **kernel**, and **initramfs** files to your HTTP server.



IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.
6. Configure PXE or iPXE installation for the RHCOS images and begin the installation.

Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE (**x86_64**):

```

DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
  APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3

```

- 1 Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **APPEND** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.

- For iPXE (**x86_64 + aarch64**):

```

kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot

```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP server. The **kernel** parameter value is the location of the **kernel** file, the **initrd=main** argument is needed for booting on UEFI systems, the **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file.

- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your HTTP server.



NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more **console=** arguments to the **kernel** line. For example, add **console=tty0 console=ttyS0** to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see [How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?](#) and "Enabling the serial console for PXE and ISO installation" in the "Advanced RHCOS installation configuration" section.



NOTE

To network boot the CoreOS **kernel** on **aarch64** architecture, you need to use a version of iPXE build with the **IMAGE_GZIP** option enabled. See [IMAGE_GZIP option in iPXE](#).

- For PXE (with UEFI and Grub as second stage) on **aarch64**:

```

menuentry 'Install CoreOS' {
  linux rhcos-<version>-live-kernel-<architecture>
  coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
  <architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
  initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}

```

- 1 Specify the locations of the RHCOS files that you uploaded to your HTTP/TFTP server. The **kernel** parameter value is the location of the **kernel** file on your TFTP server. The **coreos.live.rootfs_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file on your HTTP Server.
- 2 If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named **eno1**, set **ip=eno1:dhcp**.
- 3 Specify the location of the **initramfs** file that you uploaded to your TFTP server.

7. Monitor the progress of the RHCOS installation on the console of the machine.



IMPORTANT

Be sure that the installation is successful on each node before commencing with the OpenShift Container Platform installation. Observing the installation process can also help to determine the cause of RHCOS installation issues that might arise.

8. After RHCOS installs, the system reboots. During reboot, the system applies the Ignition config file that you specified.
9. Check the console output to verify that Ignition ran.

Example command

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. Continue to create the machines for your cluster.



IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, also create at least two compute machines before you install the cluster.

If the required network, DNS, and load balancer infrastructure are in place, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS nodes have rebooted.



NOTE

RHCOS nodes do not include a default password for the **core** user. You can access the nodes by running **ssh core@<node>.<cluster_name>.<base_domain>** as a user with access to the SSH private key that is paired to the public key that you specified in your **install_config.yaml** file. OpenShift Container Platform 4 cluster nodes running RHCOS are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes by using SSH is not recommended. However, when investigating installation issues, if the OpenShift Container Platform API is not available, or the kubelet is not properly functioning on a target node, SSH access might be required for debugging or disaster recovery.

25.1.11.3. Advanced RHCOS installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
- Customizing a live ISO or PXE boot image

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

25.1.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

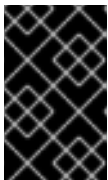
- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

Procedure

1. Boot the ISO installer.
2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as **nmcli** or **nmtui**.
3. Run the **coreos-installer** command to install the system, adding the **--copy-network** option to copy networking configuration. For example:

```
$ sudo coreos-installer install --copy-network \
  --ignition-url=http://host/worker.ign /dev/sda
```



IMPORTANT

The **--copy-network** option only copies networking configuration found under **/etc/NetworkManager/system-connections**. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

Additional resources

- See [Getting started with nmcli](#) and [Getting started with nmtui](#) in the RHEL 8 documentation for more information about the **nmcli** and **nmtui** tools.

25.1.11.3.2. Disk partitioning

Disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless you override the default partitioning configuration. During the RHCOS installation, the size of the root file system is increased to use any remaining available space on the target device.



IMPORTANT

The use of a custom partition scheme on your node might result in OpenShift Container Platform not monitoring or alerting on some node partitions. If you override the default partitioning, see [Understanding OpenShift File System Monitoring \(eviction conditions\)](#) for more information about how OpenShift Container Platform monitors your host file systems.

OpenShift Container Platform monitors the following two filesystem identifiers:

- **nodefs**, which is the filesystem that contains `/var/lib/kubelet`
- **imagefs**, which is the filesystem that contains `/var/lib/containers`

For the default partition scheme, **nodefs** and **imagefs** monitor the same root filesystem, `/`.

To override the default partitioning when installing RHCOS on an OpenShift Container Platform cluster node, you must create separate partitions.



IMPORTANT

For disk sizes larger than 100GB, and especially disk sizes larger than 1TB, create a separate `/var` partition. See "Creating a separate `/var` partition" and this [Red Hat Knowledgebase article](#) for more information.

Consider a situation where you want to add a separate storage partition for your containers and container images. For example, by mounting `/var/lib/containers` in a separate partition, the kubelet separately monitors `/var/lib/containers` as the **imagefs** directory and the root file system as the **nodefs** directory.



IMPORTANT

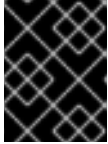
If you have resized your disk size to host a larger file system, consider creating a separate `/var/lib/containers` partition. Consider resizing a disk that has an **xfs** format to reduce CPU time issues caused by a high number of allocation groups.

25.1.11.3.2.1. Creating a separate `/var` partition

In general, you should use the default disk partitioning that is created during the RHCOS installation. However, there are cases where you might want to create a separate partition for a directory that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` directory or a subdirectory of `/var`. For example:

- **`/var/lib/containers`**: Holds container-related content that can grow as more images and containers are added to a system.
- **`/var/lib/etcd`**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **`/var`**: Holds data that you might want to keep separate for purposes such as auditing.



IMPORTANT

For disk sizes larger than 100GB, and especially larger than 1TB, create a separate **/var** partition.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

The use of a separate partition for the **/var** directory or a subdirectory of **/var** also prevents data growth in the partitioned directory from filling up the root file system.

The following procedure sets up a separate **/var** partition by adding a machine config manifest that is wrapped into the Ignition config file for a node type during the preparation phase of an installation.

Procedure

1. On your installation host, change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ openshift-install create manifests --dir <installation_directory>
```

2. Create a Butane config that configures the additional partition. For example, name the file **\$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.12.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
    - device: /dev/<device_name> 1
  partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

- 1 The storage device name of the disk that you want to partition.
- 2 When adding a data partition to the boot disk, a minimum offset value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no offset value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future

reinstalls of RHCOS might overwrite the beginning of the data partition.

- 3 The size of the data partition in mebibytes.
- 4 The **prjquota** mount option must be enabled for filesystems used for container storage.



NOTE

When creating a separate **/var** partition, you cannot use different instance types for compute nodes, if the different instance types do not have the same device name.

3. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. Create the Ignition config files:

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

- 1 For **<installation_directory>**, specify the same installation directory.

Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

The files in the **<installation_directory>/manifest** and **<installation_directory>/openshift** directories are wrapped into the Ignition config files, including the file that contains the **98-var-partition** custom **MachineConfig** object.

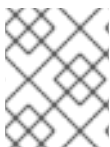
Next steps

- You can apply the custom disk partitioning by referencing the Ignition config files during the RHCOS installations.

25.1.11.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the **coreos-installer** command that cause the installer to maintain one or more existing partitions. For a PXE installation, you can add **coreos.inst.*** options to the **APPEND** parameter to preserve partitions.

Saved partitions might be data partitions from an existing OpenShift Container Platform system. You can identify the disk partitions you want to keep either by partition label or by number.



NOTE

If you save existing partitions, and those partitions do not leave enough space for RHCOS, the installation will fail without damaging the saved partitions.

Retaining existing partitions during an ISO installation

This example preserves any partition in which the partition label begins with **data** (**data***):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the **coreos-installer** in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
  --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
  --save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, **coreos-installer** recreates the partition immediately.

Retaining existing partitions during a PXE installation

This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

This **APPEND** option preserves partition 6:

```
coreos.inst.save_partindex=6
```

25.1.11.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config:** Every manual RHCOS installation needs to pass one of the Ignition config files generated by **openshift-installer**, such as **bootstrap.ign**, **master.ign** and **worker.ign**, to carry out the installation.



IMPORTANT

It is not recommended to modify these Ignition config files directly. You can update the manifest files that are wrapped into the Ignition config files, as outlined in examples in the preceding sections.

For PXE installations, you pass the Ignition configs on the **APPEND** line using the **coreos.inst.ignition_url=** option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the **coreos-installer** command line with the **--ignition-url=** option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config:** This type can be created by using the **coreos-installer customize** subcommand and its various options. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be done once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

For PXE or ISO boots, you can create the Ignition config and **APPEND** the **ignition.config.url=** option to identify the location of the Ignition config. You also need to append **ignition.firstboot** or the **ignition.platform.id=metal** option will be ignored.

25.1.11.3.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the **coreos-installer** command.

25.1.11.3.4.1. Networking and bonding options for ISO installations

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot the image to configure networking for a node. If no networking arguments are specified, DHCP is activated in the initramfs when RHCOS detects that networking is required to fetch the Ignition config file.



IMPORTANT

When adding networking arguments manually, you must also add the **rd.neednet=1** kernel argument to bring the network up in the initramfs.

The following information provides examples for configuring networking and bonding on your RHCOS nodes for ISO installations. The examples describe how to use the **ip=**, **nameserver=**, and **bond=** kernel arguments.



NOTE

Ordering is important when adding the kernel arguments: **ip=**, **nameserver=**, and then **bond=**.

The networking options are passed to the **dracut** tool during system boot. For more information about the networking options supported by **dracut**, see the [dracut.cmdline manual page](#).

The following examples are the networking options for ISO installation.

Configuring DHCP or static IP addresses

To configure an IP address, either use DHCP (**ip=dhcp**) or set an individual static IP address (**ip=<host_ip>**). If setting a static IP, you must then identify the DNS server IP address (**nameserver=<dns_ip>**) on each node. The following example sets:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The hostname to **core0.example.com**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



NOTE

When you use DHCP to configure IP addressing for the RHCOS machines, the machines also obtain the DNS server information through DHCP. For DHCP-based deployments, you can define the DNS server address that is used by the RHCOS nodes through your DHCP server configuration.

Configuring an IP address without a static hostname

You can configure an IP address without assigning a static hostname. If a static hostname is not set by the user, it will be picked up and automatically set by a reverse DNS lookup. To configure an IP address without a static hostname refer to the following example:

- The node's IP address to **10.10.10.2**
- The gateway address to **10.10.10.254**
- The netmask to **255.255.255.0**
- The DNS server address to **4.4.4.41**
- The auto-configuration value to **none**. No auto-configuration is required when IP networking is configured statically.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

Specifying multiple network interfaces

You can specify multiple network interfaces by setting multiple **ip=** entries.

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring default gateway and route

Optional: You can configure routes to additional networks by setting an **rd.route=** value.



NOTE

When you configure one or multiple networks, one default gateway is required. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.

- Run the following command to configure the default gateway:

```
ip=::10.10.10.254:::
```

- Enter the following command to configure the route for the additional network:

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

Disabling DHCP on a single interface

You can disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. In the example, the **enp1s0** interface has a static networking configuration and DHCP is disabled for **enp2s0**, which is not used:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=:::core0.example.com:enp2s0:none
```

Combining DHCP and static IP configurations

You can combine DHCP and static IP configurations on systems with multiple network interfaces, for example:

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

Configuring VLANs on individual interfaces

Optional: You can configure VLANs on individual interfaces by using the **vlan=** parameter.

- To configure a VLAN on a network interface and use a static IP address, run the following command:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- To configure a VLAN on a network interface and to use DHCP, run the following command:

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

Providing multiple DNS servers

You can provide multiple DNS servers by adding a **nameserver=** entry for each server, for example:

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

Bonding multiple network interfaces to a single interface

Optional: You can bond multiple network interfaces to a single interface by using the **bond=** option. Refer to the following examples:

- The syntax for configuring a bonded interface is: **bond=name[:network_interfaces][:options]** *name* is the bonding device name (**bond0**), *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1,em2**), and *options* is a comma-separated list of bonding options. Enter **modinfo bonding** to see available options.
- When you create a bonded interface using **bond=**, you must specify how the IP address is assigned and other information for the bonded interface.
- To configure the bonded interface to use DHCP, set the bond's IP address to **dhcp**. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

Bonding multiple network interfaces to a single interface

Optional: You can configure VLANs on bonded interfaces by using the **vlan=** parameter and to use DHCP, for example:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

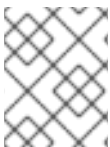
Use the following example to configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Using network teaming

Optional: You can use a network teaming as an alternative to bonding by using the **team=** parameter:

- The syntax for configuring a team interface is: **team=name[:network_interfaces]** *name* is the team device name (**team0**) and *network_interfaces* represents a comma-separated list of physical (ethernet) interfaces (**em1, em2**).



NOTE

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).

Use the following example to configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```


25.1.11.3.4.2. coreos-installer options for ISO and PXE installations

You can install RHCOS by running **coreos-installer install <options> <device>** at the command prompt, after booting into the RHCOS live environment from an ISO image.

The following table shows the subcommands, options, and arguments you can pass to the **coreos-installer** command.


Table 25.9. coreos-installer subcommands, command-line options, and arguments

coreos-installer install subcommand	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer install <options> <device>	Embed an Ignition config in an ISO image.
coreos-installer install subcommand options	
<i>Option</i>	<i>Description</i>
-u, --image-url <url>	Specify the image URL manually.
-f, --image-file <path>	Specify a local image file manually. Used for debugging.
-i, --ignition-file <path>	Embed an Ignition config from a file.
-l, --ignition-url <URL>	Embed an Ignition config from a URL.
--ignition-hash <digest>	Digest type-value of the Ignition config.
-p, --platform <name>	Override the Ignition platform ID for the installed system.
--console <spec>	Set the kernel and bootloader console for the installed system. For more information about the format of <spec> , see the Linux kernel serial console documentation.
--append-karg <arg>...	Append a default kernel argument to the installed system.
--delete-karg <arg>...	Delete a default kernel argument from the installed system.

-n, --copy-network	Copy the network configuration from the install environment.  IMPORTANT The --copy-network option only copies networking configuration found under /etc/NetworkManager/system-connections . In particular, it does not copy the system hostname.
--network-dir <path>	For use with -n . Default is /etc/NetworkManager/system-connections/ .
--save-partlabel <lx>..	Save partitions with this label glob.
--save-partindex <id>...	Save partitions with this number or range.
--insecure	Skip RHCOS image signature verification.
--insecure-ignition	Allow Ignition URL without HTTPS or hash.
--architecture <name>	Target CPU architecture. Valid values are x86_64 and aarch64 .
--preserve-on-error	Do not clear partition table on error.
-h, --help	Print help information.
coreos-installer install subcommand argument	
<i>Argument</i>	<i>Description</i>
<device>	The destination device.
coreos-installer ISO subcommands	
<i>Subcommand</i>	<i>Description</i>
\$ coreos-installer iso customize <options> <ISO_image>	Customize a RHCOS live ISO image.

coreos-installer iso reset <options> <ISO_image>	Restore a RHCOS live ISO image to default settings.
coreos-installer iso ignition remove <options> <ISO_image>	Remove the embedded Ignition config from an ISO image.
coreos-installer ISO customize subcommand options	
<i>Option</i>	<i>Description</i>
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--dest-karg-append <arg>	Add a kernel argument to each boot of the destination system.
--dest-karg-delete <arg>	Delete a kernel argument from each boot of the destination system.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
--post-install <path>	Run the specified script after installation.
--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
--live-karg-append <arg>	Add a kernel argument to each boot of the live environment.
--live-karg-delete <arg>	Delete a kernel argument from each boot of the live environment.

--live-karg-replace <k=o=n>	Replace a kernel argument in each boot of the live environment, in the form key=old=new .
-f, --force	Overwrite an existing Ignition config.
-o, --output <path>	Write the ISO to a new output file.
-h, --help	Print help information.
coreos-installer PXE subcommands	
<i>Subcommand</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
coreos-installer pxe customize <options> <path>	Customize a RHCOS live PXE boot config.
coreos-installer pxe ignition wrap <options>	Wrap an Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <image_name>	Show the wrapped Ignition config in an image.
coreos-installer PXE customize subcommand options	
<i>Option</i>	<i>Description</i>
Note that not all of these options are accepted by all subcommands.	
--dest-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the destination system.
--dest-console <spec>	Specify the kernel and bootloader console for the destination system.
--dest-device <path>	Install and overwrite the specified destination device.
--network-keyfile <path>	Configure networking by using the specified NetworkManager keyfile for live and destination systems.
--ignition-ca <path>	Specify an additional TLS certificate authority to be trusted by Ignition.
--pre-install <path>	Run the specified script before installation.
post-install <path>	Run the specified script after installation.

--installer-config <path>	Apply the specified installer configuration file.
--live-ignition <path>	Merge the specified Ignition config file into a new configuration fragment for the live environment.
-o, --output <path>	Write the initramfs to a new output file.  NOTE This option is required for PXE environments.
-h, --help	Print help information.

25.1.11.3.4.3. coreos.inst boot options for ISO or PXE installations

You can automatically invoke **coreos-installer** options at boot time by passing **coreos.inst** boot arguments to the RHCOS live installer. These are provided in addition to the standard boot arguments.

- For ISO installations, the **coreos.inst** options can be added by interrupting the automatic boot at the bootloader menu. You can interrupt the automatic boot by pressing **TAB** while the **RHEL CoreOS (Live)** menu option is highlighted.
- For PXE or iPXE installations, the **coreos.inst** options must be added to the **APPEND** line before the RHCOS live installer is booted.

The following table shows the RHCOS live installer **coreos.inst** boot options for ISO and PXE installations.

Table 25.10. **coreos.inst** boot options

Argument	Description
coreos.inst.install_dev	Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda , although sda is allowed.
coreos.inst.ignition_url	Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded. Only HTTP and HTTPS protocols are supported.
coreos.inst.save_partlabel	Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.
coreos.inst.save_partindex	Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.

Argument	Description
coreos.inst.insecure	Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.
coreos.inst.image_url	Optional: Download and install the specified RHCOS image. <ul style="list-style-type: none"> • This argument should not be used in production environments and is intended for debugging purposes only. • While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install. • If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform. • Only HTTP and HTTPS protocols are supported.
coreos.inst.skip_reboot	Optional: The system will not reboot after installing. After the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.
coreos.inst.platform_id	Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal . This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: coreos.inst.platform_id=vmware .
ignition.config.url	Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from coreos.inst.ignition_url , which is the Ignition config for the installed system.

25.1.12. Waiting for the bootstrap process to complete

The OpenShift Container Platform bootstrap process begins after the cluster nodes first boot into the persistent RHCOS environment that has been installed to disk. The configuration information provided through the Ignition config files is used to initialize the bootstrap process and install OpenShift Container Platform on the machines. You must wait for the bootstrap process to complete.

Prerequisites

- You have created the Ignition config files for your cluster.
- You have configured suitable network, DNS and load balancing infrastructure.
- You have obtained the installation program and generated the Ignition config files for your cluster.
- You installed RHCOS on your cluster machines and provided the Ignition config files that the OpenShift Container Platform installation program generated.
- Your machines have direct internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

- 1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- 2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.25.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After the bootstrap process is complete, remove the bootstrap machine from the load balancer.



IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the bootstrap machine itself.

25.1.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

- Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

- Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

25.1.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

- Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.25.0
master-1  Ready    master   63m   v1.25.0
master-2  Ready    master   64m   v1.25.0
```

The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

- Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper  Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

- If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:



NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{\n"}\n{{end}}\n{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

- Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME      AGE   REQUESTOR                                     CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

- 1** **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{"\n"}\n{{end}}{{end}}' | xargs oc adm certificate approve
```

- After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master 73m  v1.25.0
master-1  Ready   master 73m  v1.25.0
master-2  Ready   master 74m  v1.25.0
worker-0  Ready   worker 11m  v1.25.0
worker-1  Ready   worker 11m  v1.25.0
```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

25.1.15. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m
openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

2. Configure the Operators that are not available.

25.1.15.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

25.1.15.2. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**. When this has completed, you must configure storage.

25.1.15.3. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

25.1.15.3.1. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have a cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.

- You have provisioned persistent storage for your cluster, such as Red Hat OpenShift Data Foundation.



IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. **ReadWriteOnce** access also requires that the registry uses the **Recreate** rollout strategy. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

Procedure

- To configure your registry to use storage, change the **spec.storage.pvc** in the **configs.imageregistry/cluster** resource.



NOTE

When you use shared storage, review your security settings to prevent outside access.

- Verify that you do not have a registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

Example output

```
No resources found in openshift-image-registry namespace
```



NOTE

If you do have a registry pod in your output, you do not need to continue with this procedure.

- Check the registry configuration:

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Example output

```
storage:
  pvc:
    claim:
```

Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

- Check the **clusteroperator** status:

```
$ oc get clusteroperator image-registry
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
image-registry	4.12	True	False	False	6h50m

5. Ensure that your registry is set to managed to enable building and pushing of images.

- Run:

```
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```
managementState: Removed
```

to

```
managementState: Managed
```

25.1.15.3.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":{}}}'
```



WARNING

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

25.1.15.3.3. Configuring block registry storage for bare metal

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.



IMPORTANT

Block storage volumes, or block persistent volumes, are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

If you choose to use a block storage volume with the image registry, you must use a filesystem persistent volume claim (PVC).

Procedure

1. Enter the following command to set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy, and runs with only one (**1**) replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
 - a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

- 1 A unique name that represents the **PersistentVolumeClaim** object.
- 2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
- 3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
- 4 The size of the persistent volume claim.

- b. Enter the following command to create the **PersistentVolumeClaim** object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Enter the following command to edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

Example output

```
storage:
  pvc:
    claim: 1
```

- 1 By creating a custom PVC, you can leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

25.1.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.12.0	True	False	False	19m
baremetal	4.12.0	True	False	False	37m
cloud-credential	4.12.0	True	False	False	40m
cluster-autoscaler	4.12.0	True	False	False	37m
config-operator	4.12.0	True	False	False	38m
console	4.12.0	True	False	False	26m
csi-snapshot-controller	4.12.0	True	False	False	37m
dns	4.12.0	True	False	False	37m
etcd	4.12.0	True	False	False	36m
image-registry	4.12.0	True	False	False	31m
ingress	4.12.0	True	False	False	30m
insights	4.12.0	True	False	False	31m
kube-apiserver	4.12.0	True	False	False	26m
kube-controller-manager	4.12.0	True	False	False	36m
kube-scheduler	4.12.0	True	False	False	36m
kube-storage-version-migrator	4.12.0	True	False	False	37m
machine-api	4.12.0	True	False	False	29m
machine-approver	4.12.0	True	False	False	37m
machine-config	4.12.0	True	False	False	36m
marketplace	4.12.0	True	False	False	37m
monitoring	4.12.0	True	False	False	29m
network	4.12.0	True	False	False	38m
node-tuning	4.12.0	True	False	False	37m

openshift-apiserver	4.12.0	True	False	False	32m
openshift-controller-manager	4.12.0	True	False	False	30m
openshift-samples	4.12.0	True	False	False	32m
operator-lifecycle-manager	4.12.0	True	False	False	37m
operator-lifecycle-manager-catalog	4.12.0	True	False	False	37m
operator-lifecycle-manager-packageserver	4.12.0	True	False	False	32m
service-ca	4.12.0	True	False	False	38m
storage	4.12.0	True	False	False	37m

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

- 1** For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

- a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

```

NAMESPACE          NAME                                     READY STATUS
RESTARTS  AGE
openshift-apiserver-operator  openshift-apiserver-operator-85cb746d55-zqhs8  1/1
Running    1      9m
```

```

openshift-apiserver      apiserver-67b9g          1/1   Running   0
3m
openshift-apiserver      apiserver-ljcmx          1/1   Running   0
1m
openshift-apiserver      apiserver-z25h4          1/1   Running   0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8 1/1
Running   0      5m
...

```

- b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace> 1
```

- 1** Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. For an installation with Fibre Channel Protocol (FCP), additional steps are required to enable multipathing. Do not enable multipathing during installation. See "Enabling multipathing with kernel arguments on RHCOS" in the *Post-installation machine configuration tasks* documentation for more information.

25.1.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.12, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager Hybrid Cloud Console](#).

After you confirm that your [OpenShift Cluster Manager Hybrid Cloud Console](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

25.1.18. Next steps

- [Customize your cluster](#).
- If necessary, you can [opt out of remote health reporting](#) .
- [Set up your registry and configure registry storage](#) .

CHAPTER 26. INSTALLATION CONFIGURATION

26.1. CUSTOMIZING NODES

OpenShift Container Platform supports both cluster-wide and per-machine configuration via Ignition, which allows arbitrary partitioning and file content changes to the operating system. In general, if a configuration file is documented in Red Hat Enterprise Linux (RHEL), then modifying it via Ignition is supported.

There are two ways to deploy machine config changes:

- Creating machine configs that are included in manifest files to start up a cluster during **openshift-install**.
- Creating machine configs that are passed to running OpenShift Container Platform nodes via the Machine Config Operator.

Additionally, modifying the reference config, such as the Ignition config that is passed to **coreos-installer** when installing bare-metal nodes allows per-machine configuration. These changes are currently not visible to the Machine Config Operator.

The following sections describe features that you might want to configure on your nodes in this way.

26.1.1. Creating machine configs with Butane

Machine configs are used to configure control plane and worker machines by instructing machines how to create users and file systems, set up the network, install systemd units, and more.

Because modifying machine configs can be difficult, you can use Butane configs to create machine configs for you, thereby making node configuration much easier.

26.1.1.1. About Butane

Butane is a command-line utility that OpenShift Container Platform uses to provide convenient, short-hand syntax for writing machine configs, as well as for performing additional validation of machine configs. The format of the Butane config file that Butane accepts is defined in the [OpenShift Butane config spec](#).

26.1.1.2. Installing Butane

You can install the Butane tool (**butane**) to create OpenShift Container Platform machine configs from a command-line interface. You can install **butane** on Linux, Windows, or macOS by downloading the corresponding binary file.

TIP

Butane releases are backwards-compatible with older releases and with the Fedora CoreOS Config Transpiler (FCCT).

Procedure

1. Navigate to the Butane image download page at <https://mirror.openshift.com/pub/openshift-v4/clients/butane/>.

2. Get the **butane** binary:

- a. For the newest version of Butane, save the latest **butane** image to your current directory:

```
$ curl https://mirror.openshift.com/pub/openshift-v4/clients/butane/latest/butane --output butane
```

- b. Optional: For a specific type of architecture you are installing Butane on, such as aarch64 or ppc64le, indicate the appropriate URL. For example:

```
$ curl https://mirror.openshift.com/pub/openshift-v4/clients/butane/latest/butane-aarch64 --output butane
```

3. Make the downloaded binary file executable:

```
$ chmod +x butane
```

4. Move the **butane** binary file to a directory on your **PATH**.

To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification steps

- You can now use the Butane tool by running the **butane** command:

```
$ butane <butane_file>
```

26.1.1.3. Creating a MachineConfig object by using Butane

You can use Butane to produce a **MachineConfig** object so that you can configure worker or control plane nodes at installation time or via the Machine Config Operator.

Prerequisites

- You have installed the **butane** utility.

Procedure

1. Create a Butane config file. The following example creates a file named **99-worker-custom.bu** that configures the system console to show kernel debug messages and specifies custom settings for the chrony time service:

```
variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-custom
  labels:
    machineconfiguration.openshift.io/role: worker
openshift:
  kernel_arguments:
    - loglevel=7
storage:
```

```
files:
- path: /etc/chrony.conf
  mode: 0644
  overwrite: true
  contents:
    inline: |
      pool 0.rhel.pool.ntp.org iburst
      driftfile /var/lib/chrony/drift
      makestep 1.0 3
      rtcsync
      logdir /var/log/chrony
```



NOTE

The **99-worker-custom.bu** file is set to create a machine config for worker nodes. To deploy on control plane nodes, change the role from **worker** to **master**. To do both, you could repeat the whole procedure using different file names for the two types of deployments.

2. Create a **MachineConfig** object by giving Butane the file that you created in the previous step:

```
$ butane 99-worker-custom.bu -o ./99-worker-custom.yaml
```

A **MachineConfig** object YAML file is created for you to finish configuring your machines.

3. Save the Butane config in case you need to update the **MachineConfig** object in the future.
4. If the cluster is not running yet, generate manifest files and add the **MachineConfig** object YAML file to the **openshift** directory. If the cluster is already running, apply the file as follows:

```
$ oc create -f 99-worker-custom.yaml
```

Additional resources

- [Adding kernel modules to nodes](#)
- [Encrypting and mirroring disks during installation](#)

26.1.2. Adding day-1 kernel arguments

Although it is often preferable to modify kernel arguments as a day-2 activity, you might want to add kernel arguments to all master or worker nodes during initial cluster installation. Here are some reasons you might want to add kernel arguments during cluster installation so they take effect before the systems first boot up:

- You need to do some low-level network configuration before the systems start.
- You want to disable a feature, such as SELinux, so it has no impact on the systems when they first come up.

**WARNING**

Disabling SELinux on RHCOS in production is not supported. Once SELinux has been disabled on a node, it must be re-provisioned before re-inclusion in a production cluster.

To add kernel arguments to master or worker nodes, you can create a **MachineConfig** object and inject that object into the set of manifest files used by Ignition during cluster setup.

For a listing of arguments you can pass to a RHEL 8 kernel at boot time, see [Kernel.org kernel parameters](#). It is best to only add kernel arguments with this procedure if they are needed to complete the initial OpenShift Container Platform installation.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. Decide if you want to add kernel arguments to worker or control plane nodes.
3. In the **openshift** directory, create a file (for example, **99-openshift-machineconfig-master-kargs.yaml**) to define a **MachineConfig** object to add the kernel settings. This example adds a **loglevel=7** kernel argument to control plane nodes:

```
$ cat << EOF > 99-openshift-machineconfig-master-kargs.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-openshift-machineconfig-master-kargs
spec:
  kernelArguments:
    - loglevel=7
EOF
```

You can change **master** to **worker** to add kernel arguments to worker nodes instead. Create a separate YAML file to add to both master and worker nodes.

You can now continue on to create the cluster.

26.1.3. Adding kernel modules to nodes

For most common hardware, the Linux kernel includes the device driver modules needed to use that hardware when the computer starts up. For some hardware, however, modules are not available in Linux. Therefore, you must find a way to provide those modules to each host computer. This procedure describes how to do that for nodes in an OpenShift Container Platform cluster.

When a kernel module is first deployed by following these instructions, the module is made available for the current kernel. If a new kernel is installed, the `kmods-via-containers` software will rebuild and deploy the module so a compatible version of that module is available with the new kernel.

The way that this feature is able to keep the module up to date on each node is by:

- Adding a `systemd` service to each node that starts at boot time to detect if a new kernel has been installed and
- If a new kernel is detected, the service rebuilds the module and installs it to the kernel

For information on the software needed for this procedure, see the [kmods-via-containers](#) github site.

A few important issues to keep in mind:

- This procedure is Technology Preview.
- Software tools and examples are not yet available in official RPM form and can only be obtained for now from unofficial **github.com** sites noted in the procedure.
- Third-party kernel modules you might add through these procedures are not supported by Red Hat.
- In this procedure, the software needed to build your kernel modules is deployed in a RHEL 8 container. Keep in mind that modules are rebuilt automatically on each node when that node gets a new kernel. For that reason, each node needs access to a **yum** repository that contains the kernel and related packages needed to rebuild the module. That content is best provided with a valid RHEL subscription.

26.1.3.1. Building and testing the kernel module container

Before deploying kernel modules to your OpenShift Container Platform cluster, you can test the process on a separate RHEL system. Gather the kernel module's source code, the KVC framework, and the `kmod-via-containers` software. Then build and test the module. To do that on a RHEL 8 system, do the following:

Procedure

1. Register a RHEL 8 system:

```
# subscription-manager register
```

2. Attach a subscription to the RHEL 8 system:

```
# subscription-manager attach --auto
```

3. Install software that is required to build the software and container:

```
# yum install podman make git -y
```

4. Clone the **kmod-via-containers** repository:

- a. Create a folder for the repository:

```
$ mkdir kmods; cd kmods
```

- b. Clone the repository:

```
$ git clone https://github.com/kmods-via-containers/kmods-via-containers
```

5. Install a KVC framework instance on your RHEL 8 build host to test the module. This adds a **kmods-via-container** systemd service and loads it:

- a. Change to the **kmod-via-containers** directory:

```
$ cd kmods-via-containers/
```

- b. Install the KVC framework instance:

```
$ sudo make install
```

- c. Reload the systemd manager configuration:

```
$ sudo systemctl daemon-reload
```

6. Get the kernel module source code. The source code might be used to build a third-party module that you do not have control over, but is supplied by others. You will need content similar to the content shown in the **kvc-simple-kmod** example that can be cloned to your system as follows:

```
$ cd .. ; git clone https://github.com/kmods-via-containers/kvc-simple-kmod
```

7. Edit the configuration file, **simple-kmod.conf** file, in this example, and change the name of the Dockerfile to **Dockerfile.rhel**:

- a. Change to the **kvc-simple-kmod** directory:

```
$ cd kvc-simple-kmod
```

- b. Rename the Dockerfile:

```
$ cat simple-kmod.conf
```

Example Dockerfile

```
KMOD_CONTAINER_BUILD_CONTEXT="https://github.com/kmods-via-containers/kvc-  
simple-kmod.git"  
KMOD_CONTAINER_BUILD_FILE=Dockerfile.rhel  
KMOD_SOFTWARE_VERSION=dd1a7d4  
KMOD_NAMES="simple-kmod simple-procfs-kmod"
```

8. Create an instance of **kmods-via-containers@.service** for your kernel module, **simple-kmod** in this example:

```
$ sudo make install
```

9. Enable the **kmods-via-containers@.service** instance:

```
$ sudo kmods-via-containers build simple-kmod $(uname -r)
```

10. Enable and start the systemd service:

```
$ sudo systemctl enable kmods-via-containers@simple-kmod.service --now
```

- a. Review the service status:

```
$ sudo systemctl status kmods-via-containers@simple-kmod.service
```

Example output

- `kmods-via-containers@simple-kmod.service` - Kmods Via Containers - simple-kmod
Loaded: loaded (/etc/systemd/system/kmods-via-containers@.service;
enabled; vendor preset: disabled)
Active: active (exited) since Sun 2020-01-12 23:49:49 EST; 5s ago...

11. To confirm that the kernel modules are loaded, use the **lsmod** command to list the modules:

```
$ lsmod | grep simple_
```

Example output

```
simple_procfs_kmod    16384 0
simple_kmod           16384 0
```

12. Optional. Use other methods to check that the **simple-kmod** example is working:

- Look for a "Hello world" message in the kernel ring buffer with **dmesg**:

```
$ dmesg | grep 'Hello world'
```

Example output

```
[ 6420.761332] Hello world from simple_kmod.
```

- Check the value of **simple-procfs-kmod** in **/proc**:

```
$ sudo cat /proc/simple-procfs-kmod
```

Example output

```
simple-procfs-kmod number = 0
```

- Run the **spkut** command to get more information from the module:

```
$ sudo spkut 44
```

Example output

```

KVC: wrapper simple-kmod for 4.18.0-147.3.1.el8_1.x86_64
Running userspace wrapper using the kernel module container...
+ podman run -i --rm --privileged
  simple-kmod-dd1a7d4:4.18.0-147.3.1.el8_1.x86_64 spkut 44
simple-procfs-kmod number = 0
simple-procfs-kmod number = 44

```

Going forward, when the system boots this service will check if a new kernel is running. If there is a new kernel, the service builds a new version of the kernel module and then loads it. If the module is already built, it will just load it.

26.1.3.2. Provisioning a kernel module to OpenShift Container Platform

Depending on whether or not you must have the kernel module in place when OpenShift Container Platform cluster first boots, you can set up the kernel modules to be deployed in one of two ways:

- **Provision kernel modules at cluster install time (day-1)** You can create the content as a **MachineConfig** object and provide it to **openshift-install** by including it with a set of manifest files.
- **Provision kernel modules via Machine Config Operator (day-2)** If you can wait until the cluster is up and running to add your kernel module, you can deploy the kernel module software via the Machine Config Operator (MCO).

In either case, each node needs to be able to get the kernel packages and related software packages at the time that a new kernel is detected. There are a few ways you can set up each node to be able to obtain that content.

- Provide RHEL entitlements to each node.
- Get RHEL entitlements from an existing RHEL host, from the **/etc/pki/entitlement** directory and copy them to the same location as the other files you provide when you build your Ignition config.
- Inside the Dockerfile, add pointers to a **yum** repository containing the kernel and other packages. This must include new kernel packages as they are needed to match newly installed kernels.

26.1.3.2.1. Provision kernel modules via a MachineConfig object

By packaging kernel module software with a **MachineConfig** object, you can deliver that software to worker or control plane nodes at installation time or via the Machine Config Operator.

Procedure

1. Register a RHEL 8 system:

```
# subscription-manager register
```

2. Attach a subscription to the RHEL 8 system:

```
# subscription-manager attach --auto
```

3. Install software needed to build the software:

-


```
# yum install podman make git -y
```

4. Create a directory to host the kernel module and tooling:

```
$ mkdir kmods; cd kmods
```

5. Get the **kmods-via-containers** software:

- a. Clone the **kmods-via-containers** repository:

```
$ git clone https://github.com/kmods-via-containers/kmods-via-containers
```

- b. Clone the **kvc-simple-kmod** repository:

```
$ git clone https://github.com/kmods-via-containers/kvc-simple-kmod
```

6. Get your module software. In this example, **kvc-simple-kmod** is used.

7. Create a fakeroot directory and populate it with files that you want to deliver via Ignition, using the repositories cloned earlier:

- a. Create the directory:

```
$ FAKEROOT=$(mktemp -d)
```

- b. Change to the **kmod-via-containers** directory:

```
$ cd kmods-via-containers
```

- c. Install the KVC framework instance:

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

- d. Change to the **kvc-simple-kmod** directory:

```
$ cd ../kvc-simple-kmod
```

- e. Create the instance:

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

8. Clone the fakeroot directory, replacing any symbolic links with copies of their targets, by running the following command:

```
$ cd .. && rm -rf kmod-tree && cp -Lpr ${FAKEROOT} kmod-tree
```

9. Create a Butane config file, **99-simple-kmod.bu**, that embeds the kernel module tree and enables the systemd service.



NOTE

See "Creating machine configs with Butane" for information about Butane.

```

variant: openshift
version: 4.12.0
metadata:
  name: 99-simple-kmod
  labels:
    machineconfiguration.openshift.io/role: worker 1
storage:
  trees:
    - local: kmod-tree
systemd:
  units:
    - name: kmods-via-containers@simple-kmod.service
      enabled: true

```

- 1** To deploy on control plane nodes, change **worker** to **master**. To deploy on both control plane and worker nodes, perform the remainder of these instructions once for each node type.

10. Use Butane to generate a machine config YAML file, **99-simple-kmod.yaml**, containing the files and configuration to be delivered:

```
$ butane 99-simple-kmod.bu --files-dir . -o 99-simple-kmod.yaml
```

11. If the cluster is not up yet, generate manifest files and add this file to the **openshift** directory. If the cluster is already running, apply the file as follows:

```
$ oc create -f 99-simple-kmod.yaml
```

Your nodes will start the **kmods-via-containers@simple-kmod.service** service and the kernel modules will be loaded.

12. To confirm that the kernel modules are loaded, you can log in to a node (using **oc debug node/<openshift-node>**, then **chroot /host**). To list the modules, use the **lsmod** command:

```
$ lsmod | grep simple_
```

Example output

```

simple_procfs_kmod 16384 0
simple_kmod        16384 0

```

26.1.4. Encrypting and mirroring disks during installation

During an OpenShift Container Platform installation, you can enable boot disk encryption and mirroring on the cluster nodes.

26.1.4.1. About disk encryption

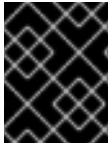
You can enable encryption for the boot disks on the control plane and compute nodes at installation time. OpenShift Container Platform supports the Trusted Platform Module (TPM) v2 and Tang encryption modes.

TPM v2

This is the preferred mode. TPM v2 stores passphrases in a secure cryptoprocessor on the server. You can use this mode to prevent decryption of the boot disk data on a cluster node if the disk is removed from the server.

Tang

Tang and Clevis are server and client components that enable network-bound disk encryption (NBDE). You can bind the boot disk data on your cluster nodes to one or more Tang servers. This prevents decryption of the data unless the nodes are on a secure network where the Tang servers are accessible. Clevis is an automated decryption framework used to implement decryption on the client side.



IMPORTANT

The use of the Tang encryption mode to encrypt your disks is only supported for bare metal and vSphere installations on user-provisioned infrastructure.

In earlier versions of Red Hat Enterprise Linux CoreOS (RHCOS), disk encryption was configured by specifying `/etc/clevis.json` in the Ignition config. That file is not supported in clusters created with OpenShift Container Platform 4.7 or later. Configure disk encryption by using the following procedure.

When the TPM v2 or Tang encryption modes are enabled, the RHCOS boot disks are encrypted using the LUKS2 format.

This feature:

- Is available for installer-provisioned infrastructure, user-provisioned infrastructure, and Assisted Installer deployments
- For Assisted installer deployments:
 - Each cluster can only have a single encryption method, Tang or TPM
 - Encryption can be enabled on some or all nodes
 - There is no Tang threshold; all servers must be valid and operational
 - Encryption applies to the installation disks only, not to the workload disks
- Is supported on Red Hat Enterprise Linux CoreOS (RHCOS) systems only
- Sets up disk encryption during the manifest installation phase, encrypting all data written to disk, from first boot forward
- Requires no user intervention for providing passphrases
- Uses AES-256-XTS encryption, or AES-256-CBC if FIPS mode is enabled

26.1.4.1.1. Configuring an encryption threshold

In OpenShift Container Platform, you can specify a requirement for more than one Tang server. You can also configure the TPM v2 and Tang encryption modes simultaneously. This enables boot disk data decryption only if the TPM secure cryptoprocessor is present and the Tang servers are accessible over a secure network.

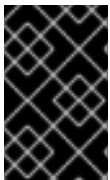
You can use the **threshold** attribute in your Butane configuration to define the minimum number of

TPM v2 and Tang encryption conditions required for decryption to occur. The threshold is met when the stated value is reached through any combination of the declared conditions. For example, the **threshold** value of **2** in the following configuration can be reached by accessing the two Tang servers, or by accessing the TPM secure cryptoprocessor and one of the Tang servers:

Example Butane configuration for disk encryption

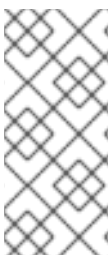
```
variant: openshift
version: 4.12.0
metadata:
  name: worker-storage
  labels:
    machineconfiguration.openshift.io/role: worker
boot_device:
  layout: x86_64 1
luks:
  tpm2: true 2
  tang: 3
    - url: http://tang1.example.com:7500
      thumbprint: jwGN5tRFK-kF6plX89ssF3khxxX
    - url: http://tang2.example.com:7500
      thumbprint: VCJsvZFjBSIHSlw78rOrq7h2ZF
  threshold: 2 4
openshift:
  fips: true
```

- 1** Set this field to the instruction set architecture of the cluster nodes. Some examples include, **x86_64**, **aarch64**, or **ppc64le**.
- 2** Include this field if you want to use a Trusted Platform Module (TPM) to encrypt the root file system.
- 3** Include this section if you want to use one or more Tang servers.
- 4** Specify the minimum number of TPM v2 and Tang encryption conditions required for decryption to occur.



IMPORTANT

The default **threshold** value is **1**. If you include multiple encryption conditions in your configuration but do not specify a threshold, decryption can occur if any of the conditions are met.



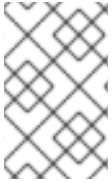
NOTE

If you require TPM v2 *and* Tang for decryption, the value of the **threshold** attribute must equal the total number of stated Tang servers plus one. If the **threshold** value is lower, it is possible to reach the threshold value by using a single encryption mode. For example, if you set **tpm2** to **true** and specify two Tang servers, a threshold of **2** can be met by accessing the two Tang servers, even if the TPM secure cryptoprocessor is not available.

26.1.4.2. About disk mirroring

During OpenShift Container Platform installation on control plane and worker nodes, you can enable mirroring of the boot and other disks to two or more redundant storage devices. A node continues to function after storage device failure provided one device remains available.

Mirroring does not support replacement of a failed disk. Reprovision the node to restore the mirror to a pristine, non-degraded state.



NOTE

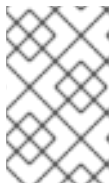
For user-provisioned infrastructure deployments, mirroring is available only on RHCOS systems. Support for mirroring is available on **x86_64** nodes booted with BIOS or UEFI and on **ppc64le** nodes.

26.1.4.3. Configuring disk encryption and mirroring

You can enable and configure encryption and mirroring during an OpenShift Container Platform installation.

Prerequisites

- You have downloaded the OpenShift Container Platform installation program on your installation node.
- You installed Butane on your installation node.



NOTE

Butane is a command-line utility that OpenShift Container Platform uses to offer convenient, short-hand syntax for writing and validating machine configs. For more information, see "Creating machine configs with Butane".

- You have access to a Red Hat Enterprise Linux (RHEL) 8 machine that can be used to generate a thumbprint of the Tang exchange key.

Procedure

1. If you want to use TPM v2 to encrypt your cluster, check to see if TPM v2 encryption needs to be enabled in the host firmware for each node. This is required on most Dell systems. Check the manual for your specific system.
2. If you want to use Tang to encrypt your cluster, follow these preparatory steps:
 - a. Set up a Tang server or access an existing one. See [Network-bound disk encryption](#) for instructions.
 - b. Install the **clevis** package on a RHEL 8 machine, if it is not already installed:

```
$ sudo yum install clevis
```

- c. On the RHEL 8 machine, run the following command to generate a thumbprint of the exchange key. Replace **http://tang.example.com:7500** with the URL of your Tang server:

```
$ clevis-encrypt-tang '{"url":"http://tang.example.com:7500"}' < /dev/null > /dev/null 1
```

- 1 In this example, **tangd.socket** is listening on port **7500** on the Tang server.



NOTE

The **clevis-encrypt-tang** command generates a thumbprint of the exchange key. No data passes to the encryption command during this step; **/dev/null** exists here as an input instead of plain text. The encrypted output is also sent to **/dev/null**, because it is not required for this procedure.

Example output

The advertisement contains the following signing keys:

```
PLjNyRdGw03zIRoGjQYMahSZGu9 1
```

- 1 The thumbprint of the exchange key.

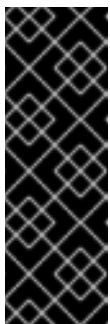
When the **Do you wish to trust these keys? [ynYN]** prompt displays, type **Y**.



NOTE

RHEL 8 provides Clevis version 15, which uses the SHA-1 hash algorithm to generate thumbprints. Some other distributions provide Clevis version 17 or later, which use the SHA-256 hash algorithm for thumbprints. You must use a Clevis version that uses SHA-1 to create the thumbprint, to prevent Clevis binding issues when you install Red Hat Enterprise Linux CoreOS (RHCOS) on your OpenShift Container Platform cluster nodes.

- d. If the nodes are configured with static IP addressing, run **coreos-installer iso customize --dest-karg-append** or use the **coreos-installer --append-karg** option when installing RHCOS nodes to set the IP address of the installed system. Append the **ip=** and other arguments needed for your network.



IMPORTANT

Some methods for configuring static IPs do not affect the initramfs after the first boot and will not work with Tang encryption. These include the **coreos-installer --copy-network** option, the **coreos-installer iso customize --network-keyfile** option, and the **coreos-installer pxe customize --network-keyfile** option, as well as adding **ip=** arguments to the kernel command line of the live ISO or PXE image during installation. Incorrect static IP configuration causes the second boot of the node to fail.

3. On your installation node, change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

- 1 Replace **<installation_directory>** with the path to the directory that you want to store the installation files in.

4. Create a Butane config that configures disk encryption, mirroring, or both. For example, to configure storage for compute nodes, create a **\$HOME/clusterconfig/worker-storage.bu** file.

Butane config example for a boot device

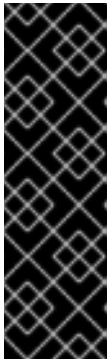
```

variant: openshift
version: 4.12.0
metadata:
  name: worker-storage ❶
  labels:
    machineconfiguration.openshift.io/role: worker ❷
boot_device:
  layout: x86_64 ❸
  luks: ❹
  tpm2: true ❺
  tang: ❻
    - url: http://tang.example.com:7500 ❼
      thumbprint: PLjNyRdGw03zIRoGjQYMahSZGu9 ❽
  threshold: 1 ❾
  mirror: ❿
  devices: ❾
    - /dev/sda
    - /dev/sdb
openshift:
  fips: true ❿

```

- ❶ ❷ For control plane configurations, replace **worker** with **master** in both of these locations.
- ❸ Set this field to the instruction set architecture of the cluster nodes. Some examples include, **x86_64**, **aarch64**, or **ppc64le**.
- ❹ Include this section if you want to encrypt the root file system. For more details, see "About disk encryption".
- ❺ Include this field if you want to use a Trusted Platform Module (TPM) to encrypt the root file system.
- ❻ Include this section if you want to use one or more Tang servers.
- ❼ Specify the URL of a Tang server. In this example, **tangd.socket** is listening on port **7500** on the Tang server.
- ❽ Specify the exchange key thumbprint, which was generated in a preceding step.
- ❾ Specify the minimum number of TPM v2 and Tang encryption conditions that must be met for decryption to occur. The default value is **1**. For more information about this topic, see "Configuring an encryption threshold".
- ❿ Include this section if you want to mirror the boot disk. For more details, see "About disk mirroring".
- ❾ List all disk devices that should be included in the boot disk mirror, including the disk that RHCOS will be installed onto.

- 12 Include this directive to enable FIPS mode on your cluster.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#) . If you are configuring nodes to use both disk encryption and mirroring, both features must be configured in the same Butane configuration file. If you are configuring disk encryption on a node with FIPS mode enabled, you must include the **fips** directive in the same Butane configuration file, even if FIPS mode is also enabled in a separate manifest.

5. Create a control plane or compute node manifest from the corresponding Butane configuration file and save it to the **<installation_directory>/openshift** directory. For example, to create a manifest for the compute nodes, run the following command:

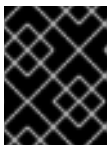
```
$ butane $HOME/clusterconfig/worker-storage.bu -o <installation_directory>/openshift/99-
worker-storage.yaml
```

Repeat this step for each node type that requires disk encryption or mirroring.

6. Save the Butane configuration file in case you need to update the manifests in the future.
7. Continue with the remainder of the OpenShift Container Platform installation.

TIP

You can monitor the console log on the RHCOS nodes during installation for error messages relating to disk encryption or mirroring.



IMPORTANT

If you configure additional data partitions, they will not be encrypted unless encryption is explicitly requested.

Verification

After installing OpenShift Container Platform, you can verify if boot disk encryption or mirroring is enabled on the cluster nodes.

1. From the installation host, access a cluster node by using a debug pod:

- a. Start a debug pod for the node, for example:

```
$ oc debug node/compute-1
```

- b. Set **/host** as the root directory within the debug shell. The debug pod mounts the root file system of the node in **/host** within the pod. By changing the root directory to **/host**, you can run binaries contained in the executable paths on the node:

```
# chroot /host
```




NOTE

OpenShift Container Platform cluster nodes running Red Hat Enterprise Linux CoreOS (RHCOS) are immutable and rely on Operators to apply cluster changes. Accessing cluster nodes using SSH is not recommended. However, if the OpenShift Container Platform API is not available, or **kubelet** is not properly functioning on the target node, **oc** operations will be impacted. In such situations, it is possible to access nodes using **ssh core@<node>.<cluster_name>.<base_domain>** instead.

2. If you configured boot disk encryption, verify if it is enabled:
 - a. From the debug shell, review the status of the root mapping on the node:

```
# cryptsetup status root
```

Example output

```
/dev/mapper/root is active and is in use.
type: LUKS2 1
cipher: aes-xts-plain64 2
keysize: 512 bits
key location: keyring
device: /dev/sda4 3
sector size: 512
offset: 32768 sectors
size: 15683456 sectors
mode: read/write
```

- 1 The encryption format. When the TPM v2 or Tang encryption modes are enabled, the RHCOS boot disks are encrypted using the LUKS2 format.
- 2 The encryption algorithm used to encrypt the LUKS2 volume. The **aes-cbc-essiv:sha256** cipher is used if FIPS mode is enabled.
- 3 The device that contains the encrypted LUKS2 volume. If mirroring is enabled, the value will represent a software mirror device, for example **/dev/md126**.

- b. List the Clevis plugins that are bound to the encrypted device:

```
# clevis luks list -d /dev/sda4 1
```

- 1 Specify the device that is listed in the **device** field in the output of the preceding step.

Example output

```
1: sss '{"t":1,"pins":{"tang":{"url":"http://tang.example.com:7500"}}}' 1
```

- 1 In the example output, the Tang plugin is used by the Shamir's Secret Sharing (SSS) Clevis plugin for the **/dev/sda4** device.

3. If you configured mirroring, verify if it is enabled:

a. From the debug shell, list the software RAID devices on the node:

```
# cat /proc/mdstat
```

Example output

```
Personalities : [raid1]
md126 : active raid1 sdb3[1] sda3[0] 1
        393152 blocks super 1.0 [2/2] [UU]

md127 : active raid1 sda4[0] sdb4[1] 2
        51869632 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

- 1** The `/dev/md126` software RAID mirror device uses the `/dev/sda3` and `/dev/sdb3` disk devices on the cluster node.
- 2** The `/dev/md127` software RAID mirror device uses the `/dev/sda4` and `/dev/sdb4` disk devices on the cluster node.

b. Review the details of each of the software RAID devices listed in the output of the preceding command. The following example lists the details of the `/dev/md126` device:

```
# mdadm --detail /dev/md126
```

Example output

```
/dev/md126:
  Version : 1.0
  Creation Time : Wed Jul 7 11:07:36 2021
  Raid Level : raid1 1
  Array Size : 393152 (383.94 MiB 402.59 MB)
  Used Dev Size : 393152 (383.94 MiB 402.59 MB)
  Raid Devices : 2
  Total Devices : 2
  Persistence : Superblock is persistent

  Update Time : Wed Jul 7 11:18:24 2021
  State : clean 2
  Active Devices : 2 3
  Working Devices : 2 4
  Failed Devices : 0 5
  Spare Devices : 0

  Consistency Policy : resync

  Name : any:md-boot 6
  UUID : ccfa3801:c520e0b5:2bee2755:69043055
  Events : 19
```

```

Number Major Minor RaidDevice State
  0   252    3    0   active sync  /dev/sda3 7
  1   252   19    1   active sync  /dev/sdb3 8

```

- 1** Specifies the RAID level of the device. **raid1** indicates RAID 1 disk mirroring.
- 2** Specifies the state of the RAID device.
- 3** **4** States the number of underlying disk devices that are active and working.
- 5** States the number of underlying disk devices that are in a failed state.
- 6** The name of the software RAID device.
- 7** **8** Provides information about the underlying disk devices used by the software RAID device.

- c. List the file systems mounted on the software RAID devices:

```
# mount | grep /dev/md
```

Example output

```

/dev/md127 on / type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /etc type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /usr type xfs
(ro,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /sysroot type xfs
(ro,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/containers/storage/overlay type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-99c050d4e0c0/volume-
subpaths/etc/tuned/1 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-99c050d4e0c0/volume-
subpaths/etc/tuned/2 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-99c050d4e0c0/volume-
subpaths/etc/tuned/3 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-99c050d4e0c0/volume-
subpaths/etc/tuned/4 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-99c050d4e0c0/volume-
subpaths/etc/tuned/5 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md126 on /boot type ext4 (rw,relatime,seclabel)

```

In the example output, the **/boot** file system is mounted on the **/dev/md126** software RAID device and the root file system is mounted on **/dev/md127**.

- Repeat the verification steps for each OpenShift Container Platform node type.

Additional resources

- For more information about the TPM v2 and Tang encryption modes, see [Configuring automated unlocking of encrypted volumes using policy-based decryption](#).

26.1.4.4. Configuring a RAID-enabled data volume

You can enable software RAID partitioning to provide an external data volume. OpenShift Container Platform supports RAID 0, RAID 1, RAID 4, RAID 5, RAID 6, and RAID 10 for data protection and fault tolerance. See "About disk mirroring" for more details.

Prerequisites

- You have downloaded the OpenShift Container Platform installation program on your installation node.
- You have installed Butane on your installation node.



NOTE

Butane is a command-line utility that OpenShift Container Platform uses to provide convenient, short-hand syntax for writing machine configs, as well as for performing additional validation of machine configs. For more information, see the *Creating machine configs with Butane* section.

Procedure

- Create a Butane config that configures a data volume by using software RAID.
 - To configure a data volume with RAID 1 on the same disks that are used for a mirrored boot disk, create a **\$HOME/clusterconfig/raid1-storage.bu** file, for example:

RAID 1 on mirrored boot disk

```
variant: openshift
version: 4.12.0
metadata:
  name: raid1-storage
  labels:
    machineconfiguration.openshift.io/role: worker
boot_device:
  mirror:
    devices:
      - /dev/sda
      - /dev/sdb
storage:
  disks:
    - device: /dev/sda
      partitions:
        - label: root-1
          size_mib: 25000 1
        - label: var-1
      - device: /dev/sdb
```

```

partitions:
  - label: root-2
    size_mib: 25000 2
  - label: var-2
raid:
  - name: md-var
    level: raid1
    devices:
      - /dev/disk/by-partlabel/var-1
      - /dev/disk/by-partlabel/var-2
filesystems:
  - device: /dev/md/md-var
    path: /var
    format: xfs
    wipe_filesystem: true
    with_mount_unit: true

```

- 1 2 When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

- To configure a data volume with RAID 1 on secondary disks, create a **\$HOME/clusterconfig/raid1-alt-storage.bu** file, for example:

RAID 1 on secondary disks

```

variant: openshift
version: 4.12.0
metadata:
  name: raid1-alt-storage
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  disks:
    - device: /dev/sdc
      wipe_table: true
      partitions:
        - label: data-1
    - device: /dev/sdd
      wipe_table: true
      partitions:
        - label: data-2
  raid:
    - name: md-var-lib-containers
      level: raid1
      devices:
        - /dev/disk/by-partlabel/data-1
        - /dev/disk/by-partlabel/data-2
  filesystems:
    - device: /dev/md/md-var-lib-containers
      path: /var/lib/containers
      format: xfs
      wipe_filesystem: true
      with_mount_unit: true

```

- 2. Create a RAID manifest from the Butane config you created in the previous step and save it to the **<installation_directory>/openshift** directory. For example, to create a manifest for the compute nodes, run the following command:

```
$ butane $HOME/clusterconfig/<butane_config>.bu -o
<installation_directory>/openshift/<manifest_name>.yaml 1
```

- 1 Replace **<butane_config>** and **<manifest_name>** with the file names from the previous step. For example, **raid1-alt-storage.bu** and **raid1-alt-storage.yaml** for secondary disks.

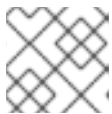
- 3. Save the Butane config in case you need to update the manifest in the future.
- 4. Continue with the remainder of the OpenShift Container Platform installation.

26.1.5. Configuring chrony time service

You can set the time server and related settings used by the chrony time service (**chronyd**) by modifying the contents of the **chrony.conf** file and passing those contents to your nodes as a machine config.

Procedure

1. Create a Butane config including the contents of the **chrony.conf** file. For example, to configure chrony on worker nodes, create a **99-worker-chrony.bu** file.



NOTE

See "Creating machine configs with Butane" for information about Butane.

```
variant: openshift
version: 4.12.0
metadata:
  name: 99-worker-chrony 1
  labels:
    machineconfiguration.openshift.io/role: worker 2
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644 3
    overwrite: true
    contents:
      inline: |
        pool 0.rhel.pool.ntp.org iburst 4
        driftfile /var/lib/chrony/drift
        makestep 1.0 3
        rtcsync
        logdir /var/log/chrony
```

- 1 2 On control plane nodes, substitute **master** for **worker** in both of these locations.

- 3 Specify an octal value mode for the **mode** field in the machine config file. After creating

4. Specify any valid, reachable time source, such as the one provided by your DHCP server. Alternately, you can specify any of the following NTP servers: **1.rhel.pool.ntp.org**,
2. Use Butane to generate a **MachineConfig** object file, **99-worker-chrony.yaml**, containing the configuration to be delivered to the nodes:

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. Apply the configurations in one of two ways:
 - If the cluster is not running yet, after you generate manifest files, add the **MachineConfig** object file to the **<installation_directory>/openshift** directory, and then continue to create the cluster.
 - If the cluster is already running, apply the file:

```
$ oc apply -f ./99-worker-chrony.yaml
```

26.1.6. Additional resources

- For information on Butane, see [Creating machine configs with Butane](#).
- For information on FIPS support, see [Support for FIPS cryptography](#).

26.2. CONFIGURING YOUR FIREWALL

If you use a firewall, you must configure it so that OpenShift Container Platform can access the sites that it requires to function. You must always grant access to some sites, and you grant access to more if you use Red Hat Insights, the Telemetry service, a cloud to host your cluster, and certain build strategies.

26.2.1. Configuring your firewall for OpenShift Container Platform

Before you install OpenShift Container Platform, you must configure your firewall to grant access to the sites that OpenShift Container Platform requires.

There are no special configuration considerations for services running on only controller nodes compared to worker nodes.



NOTE

If your environment has a dedicated load balancer in front of your OpenShift Container Platform cluster, review the allowlists between your firewall and load balancer to prevent unwanted network restrictions to your cluster.

Procedure

1. Allowlist the following registry URLs:

URL	Port	Function
registry.redhat.io	443	Provides core container images

URL	Port	Function
access.redhat.com ^[1]	443	Hosts all the container images that are stored on the Red Hat Ecosystem Catalog, including core container images.
quay.io	443	Provides core container images
cdn.quay.io	443	Provides core container images
cdn01.quay.io	443	Provides core container images
cdn02.quay.io	443	Provides core container images
cdn03.quay.io	443	Provides core container images
sso.redhat.com	443	The https://console.redhat.com site uses authentication from sso.redhat.com

1. In a firewall environment, ensure that the **access.redhat.com** resource is on the allowlist. This resource hosts a signature store that a container client requires for verifying images when pulling them from **registry.access.redhat.com**.

You can use the wildcards ***.quay.io** and ***.openshiftapps.com** instead of **cdn.quay.io** and **cdn0[1-3].quay.io** in your allowlist. When you add a site, such as **quay.io**, to your allowlist, do not add a wildcard entry, such as ***.quay.io**, to your denylist. In most cases, image registries use a content delivery network (CDN) to serve images. If a firewall blocks access, image downloads are denied when the initial download request redirects to a hostname such as **cdn01.quay.io**.

2. Allowlist any site that provides resources for a language or framework that your builds require.
3. If you do not disable Telemetry, you must grant access to the following URLs to access Red Hat Insights:

URL	Port	Function
cert-api.access.redhat.com	443	Required for Telemetry
api.access.redhat.com	443	Required for Telemetry
infogw.api.openshift.com	443	Required for Telemetry
console.redhat.com	443	Required for Telemetry and for insights-operator

4. If you use Alibaba Cloud, Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) to host your cluster, you must grant access to the URLs that provide the cloud provider API and DNS for that cloud:

Cloud	URL	Port	Function
Alibaba	*.aliyuncs.com	443	Required to access Alibaba Cloud services and resources. Review the Alibaba endpoints_config.go file to determine the exact endpoints to allow for the regions that you use.
AWS	*.amazonaws.com Alternatively, if you choose to not use a wildcard for AWS APIs, you must allowlist the following URLs:	443	Required to access AWS services and resources. Review the AWS Service Endpoints in the AWS documentation to determine the exact endpoints to allow for the regions that you use.
	ec2.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	events.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	iam.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	route53.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	*.s3.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	*.s3.<aws_region>.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	*.s3.dualstack.<aws_region>.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	sts.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	sts.<aws_region>.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	tagging.us-east-1.amazonaws.com	443	Used to install and manage clusters in an AWS environment. This endpoint is always us-east-1 , regardless of the region the cluster is deployed in.
	ec2.<aws_region>.amazonaws.com	443	Used to install and manage clusters in an AWS environment.

Cloud	URL	Port	Function
	elasticloadbalancing. <aws_region>.amazonaws.com	443	Used to install and manage clusters in an AWS environment.
	servicequotas. <aws_region>.amazonaws.com	443	Required. Used to confirm quotas for deploying the service.
	tagging. <aws_region>.amazonaws.com	443	Allows the assignment of metadata about AWS resources in the form of tags.
GCP	*.googleapis.com	443	Required to access GCP services and resources. Review Cloud Endpoints in the GCP documentation to determine the endpoints to allow for your APIs.
	accounts.google.com	443	Required to access your GCP account.
Azure	management.azure.com	443	Required to access Azure services and resources. Review the Azure REST API reference in the Azure documentation to determine the endpoints to allow for your APIs.
	*.blob.core.windows.net	443	Required to download Ignition files.
	login.microsoftonline.com	443	Required to access Azure services and resources. Review the Azure REST API reference in the Azure documentation to determine the endpoints to allow for your APIs.

5. Allowlist the following URLs:

URL	Port	Function
mirror.openshift.com	443	Required to access mirrored installation content and images. This site is also a source of release image signatures, although the Cluster Version Operator needs only a single functioning source.
storage.googleapis.com/openshift-release	443	A source of release image signatures, although the Cluster Version Operator needs only a single functioning source.

URL	Port	Function
*.apps.<cluster_name>.<base_domain>	443	Required to access the default cluster routes unless you set an ingress wildcard during installation.
quayio-production-s3.s3.amazonaws.com	443	Required to access Quay image content in AWS.
api.openshift.com	443	Required both for your cluster token and to check if updates are available for the cluster.
rhcos.mirror.openshift.com	443	Required to download Red Hat Enterprise Linux CoreOS (RHCOS) images.
console.redhat.com	443	Required for your cluster token.
sso.redhat.com	443	The https://console.redhat.com site uses authentication from sso.redhat.com

Operators require route access to perform health checks. Specifically, the authentication and web console Operators connect to two routes to verify that the routes work. If you are the cluster administrator and do not want to allow ***.apps.<cluster_name>.<base_domain>**, then allow these routes:

- **oauth-openshift.apps.<cluster_name>.<base_domain>**
- **console-openshift-console.apps.<cluster_name>.<base_domain>**, or the hostname that is specified in the **spec.route.hostname** field of the **consoles.operator/cluster** object if the field is not empty.

6. Allowlist the following URLs for optional third-party content:

URL	Port	Function
registry.connect.redhat.com	443	Required for all third-party images and certified operators.
rhc4tp-prod-z8cxf-image-registry-us-east-1-evenkyleffocxqvofrk.s3.dualstack.us-east-1.amazonaws.com	443	Provides access to container images hosted on registry.connect.redhat.com
oso-rhc4tp-docker-registry.s3-us-west-2.amazonaws.com	443	Required for Sonatype Nexus, F5 Big IP operators.

7. If you use a default Red Hat Network Time Protocol (NTP) server allow the following URLs:

- 1.rhel.pool.ntp.org
- 2.rhel.pool.ntp.org
- 3.rhel.pool.ntp.org



NOTE

If you do not use a default Red Hat NTP server, verify the NTP server for your platform and allow it in your firewall.

26.3. ENABLING LINUX CONTROL GROUP VERSION 2 (CGROUP V2)

You can enable [Linux control group version 2](#) (cgroup v2) in your cluster by editing the `node.config` object. Enabling cgroup v2 in OpenShift Container Platform disables all cgroups version 1 controllers and hierarchies in your cluster. cgroup v1 is enabled by default.

cgroup v2 is the next version of the Linux cgroup API. cgroup v2 offers several improvements over cgroup v1, including a unified hierarchy, safer sub-tree delegation, new features such as [Pressure Stall Information](#), and enhanced resource management and isolation.



IMPORTANT

OpenShift Container Platform cgroups version 2 support is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

26.3.1. Enabling Linux cgroup v2 during installation

You can enable Linux control group version 2 (cgroup v2) when you install a cluster by creating installation manifests.

Procedure

1. Create or edit the `node.config` object to specify the `v2` cgroup:

```
apiVersion: config.openshift.io/v1
kind: Node
metadata:
  name: cluster
spec:
  cgroupMode: "v1"
```

2. Create or edit the `FeatureGate` object to enable the `TechPreviewNoUpgrade` feature set:

```
apiVersion: config.openshift.io/v1
kind: FeatureGate
metadata:
```

```
name: cluster
spec:
  featureSet: "TechPreviewNoUpgrade"
```

3. Proceed with the installation as usual.

Additional resources

- [Enabling OpenShift Container Platform features using FeatureGates](#)
- [OpenShift Container Platform installation overview](#)

CHAPTER 27. VALIDATING AN INSTALLATION

You can check the status of an OpenShift Container Platform cluster after an installation by following the procedures in this document.

27.1. REVIEWING THE INSTALLATION LOG

You can review a summary of an installation in the OpenShift Container Platform installation log. If an installation succeeds, the information required to access the cluster is included in the log.

Prerequisites

- You have access to the installation host.

Procedure

- Review the `.openshift_install.log` log file in the installation directory on your installation host:

```
$ cat <install_dir>/.openshift_install.log
```

Example output

Cluster credentials are included at the end of the log if the installation is successful, as outlined in the following example:

```
...
time="2020-12-03T09:50:47Z" level=info msg="Install complete!"
time="2020-12-03T09:50:47Z" level=info msg="To access the cluster as the system:admin
user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'"
time="2020-12-03T09:50:47Z" level=info msg="Access the OpenShift web-console here:
https://console-openshift-console.apps.mycluster.example.com"
time="2020-12-03T09:50:47Z" level=info msg="Login to the console with user: \"kubeadmin\",
and password: \"password\""
time="2020-12-03T09:50:47Z" level=debug msg="Time elapsed per stage:"
time="2020-12-03T09:50:47Z" level=debug msg="  Infrastructure: 6m45s"
time="2020-12-03T09:50:47Z" level=debug msg="Bootstrap Complete: 11m30s"
time="2020-12-03T09:50:47Z" level=debug msg=" Bootstrap Destroy: 1m5s"
time="2020-12-03T09:50:47Z" level=debug msg=" Cluster Operators: 17m31s"
time="2020-12-03T09:50:47Z" level=info msg="Time elapsed: 37m26s"
```

27.2. VIEWING THE IMAGE PULL SOURCE

For clusters with unrestricted network connectivity, you can view the source of your pulled images by using a command on a node, such as `crictl images`.

However, for disconnected installations, to view the source of pulled images, you must review the CRI-O logs to locate the **Trying to access** log entry, as shown in the following procedure. Other methods to view the image pull source, such as the `crictl images` command, show the non-mirrored image name, even though the image is pulled from the mirrored location.

Prerequisites

- You have access to the cluster as a user with the `cluster-admin` role.

Procedure

- Review the CRI-O logs for a master or worker node:

```
$ oc adm node-logs <node_name> -u crio
```

Example output

The **Trying to access** log entry indicates where the image is being pulled from.

```
...
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1366]: time="2021-08-05
10:33:21.594930907Z" level=info msg="Pulling image: quay.io/openshift-release-dev/ocp-
release:4.10.0-ppc64le" id=abcd713b-d0e1-4844-ac1c-474c5b60c07c
name=/runtime.v1alpha2.ImageService/PullImage
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.194341109Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.226788351Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
...
```

The log might show the image pull source twice, as shown in the preceding example.

If your **ImageContentSourcePolicy** object lists multiple mirrors, OpenShift Container Platform attempts to pull the images in the order listed in the configuration, for example:

```
Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
Trying to access \"li0317gcp2.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
```

27.3. GETTING CLUSTER VERSION, STATUS, AND UPDATE DETAILS

You can view the cluster version and status by running the **oc get clusterversion** command. If the status shows that the installation is still progressing, you can review the status of the Operators for more information.

You can also list the current update channel and review the available cluster updates.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the cluster version and overall status:

■

```
$ oc get clusterversion
```

Example output

```
NAME     VERSION  AVAILABLE  PROGRESSING  SINCE   STATUS
version  4.6.4    True       False        6m25s  Cluster version is 4.6.4
```

The example output indicates that the cluster has been installed successfully.

2. If the cluster status indicates that the installation is still progressing, you can obtain more detailed progress information by checking the status of the Operators:

```
$ oc get clusteroperators.config.openshift.io
```

3. View a detailed summary of cluster specifications, update availability, and update history:

```
$ oc describe clusterversion
```

4. List the current update channel:

```
$ oc get clusterversion -o jsonpath='{.items[0].spec}{"\n"}'
```

Example output

```
{"channel":"stable-4.6","clusterID":"245539c1-72a3-41aa-9cec-72ed8cf25c5c"}
```

5. Review the available cluster updates:

```
$ oc adm upgrade
```

Example output

```
Cluster version is 4.6.4
```

```
Updates:
```

```
VERSION IMAGE
```

```
4.6.6 quay.io/openshift-release-dev/ocp-
release@sha256:c7e8f18e8116356701bd23ae3a23fb9892dd5ea66c8300662ef30563d7104f3
9
```

Additional resources

- See [Querying Operator status after installation](#) for more information about querying Operator status if your installation is still progressing.
- See [Troubleshooting Operator issues](#) for information about investigating issues with Operators.
- See [Updating a cluster between minor versions](#) for more information on updating your cluster.
- See [Understanding update channels and releases](#) for an overview about update release channels.

27.4. QUERYING THE STATUS OF THE CLUSTER NODES BY USING THE CLI

You can verify the status of the cluster nodes after an installation.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. List the status of the cluster nodes. Verify that the output lists all of the expected control plane and compute nodes and that each node has a **Ready** status:

```
$ oc get nodes
```

Example output

```
NAME                                STATUS  ROLES  AGE  VERSION
compute-1.example.com              Ready  worker  33m  v1.25.0
control-plane-1.example.com         Ready  master  41m  v1.25.0
control-plane-2.example.com         Ready  master  45m  v1.25.0
compute-2.example.com               Ready  worker  38m  v1.25.0
compute-3.example.com               Ready  worker  33m  v1.25.0
control-plane-3.example.com         Ready  master  41m  v1.25.0
```

2. Review CPU and memory resource availability for each cluster node:

```
$ oc adm top nodes
```

Example output

```
NAME                                CPU(cores)  CPU%  MEMORY(bytes)  MEMORY%
compute-1.example.com              128m        8%    1132Mi         16%
control-plane-1.example.com         801m        22%    3471Mi         23%
control-plane-2.example.com         1718m       49%    6085Mi         40%
compute-2.example.com               935m        62%    5178Mi         75%
compute-3.example.com               111m        7%    1131Mi         16%
control-plane-3.example.com         942m        26%    4100Mi         27%
```

Additional resources

- See [Verifying node health](#) for more details about reviewing node health and investigating node issues.

27.5. REVIEWING THE CLUSTER STATUS FROM THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE

You can review the following information in the **Overview** page in the OpenShift Container Platform web console:

- The general status of your cluster
- The status of the control plane, cluster Operators, and storage
- CPU, memory, file system, network transfer, and pod availability
- The API address of the cluster, the cluster ID, and the name of the provider
- Cluster version information
- Cluster update status, including details of the current update channel and available updates
- A cluster inventory detailing node, pod, storage class, and persistent volume claim (PVC) information
- A list of ongoing cluster activities and recent events

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

- In the **Administrator** perspective, navigate to **Home** → **Overview**.

27.6. REVIEWING THE CLUSTER STATUS FROM RED HAT OPENSIFT CLUSTER MANAGER

From the OpenShift Container Platform web console, you can review detailed information about the status of your cluster on OpenShift Cluster Manager.

Prerequisites

- You are logged in to [OpenShift Cluster Manager Hybrid Cloud Console](#).
- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Go to the **Clusters** list in [OpenShift Cluster Manager Hybrid Cloud Console](#) and locate your OpenShift Container Platform cluster.
2. Click the **Overview** tab for your cluster.
3. Review the following information about your cluster:
 - vCPU and memory availability and resource usage
 - The cluster ID, status, type, region, and the provider name
 - Node counts by node type
 - Cluster version details, the creation date of the cluster, and the name of the cluster owner
 - The life cycle support status of the cluster

- Subscription information, including the service level agreement (SLA) status, the subscription unit type, the production status of the cluster, the subscription obligation, and the service level

TIP

To view the history for your cluster, click the **Cluster history** tab.

4. Navigate to the **Monitoring** page to review the following information:
 - A list of any issues that have been detected
 - A list of alerts that are firing
 - The cluster Operator status and version
 - The cluster's resource usage
5. Optional: You can view information about your cluster that Red Hat Insights collects by navigating to the **Overview** menu. From this menu you can view the following information:
 - Potential issues that your cluster might be exposed to, categorized by risk level
 - Health-check status by category

Additional resources

- See [Using Insights to identify issues with your cluster](#) for more information about reviewing potential issues with your cluster.

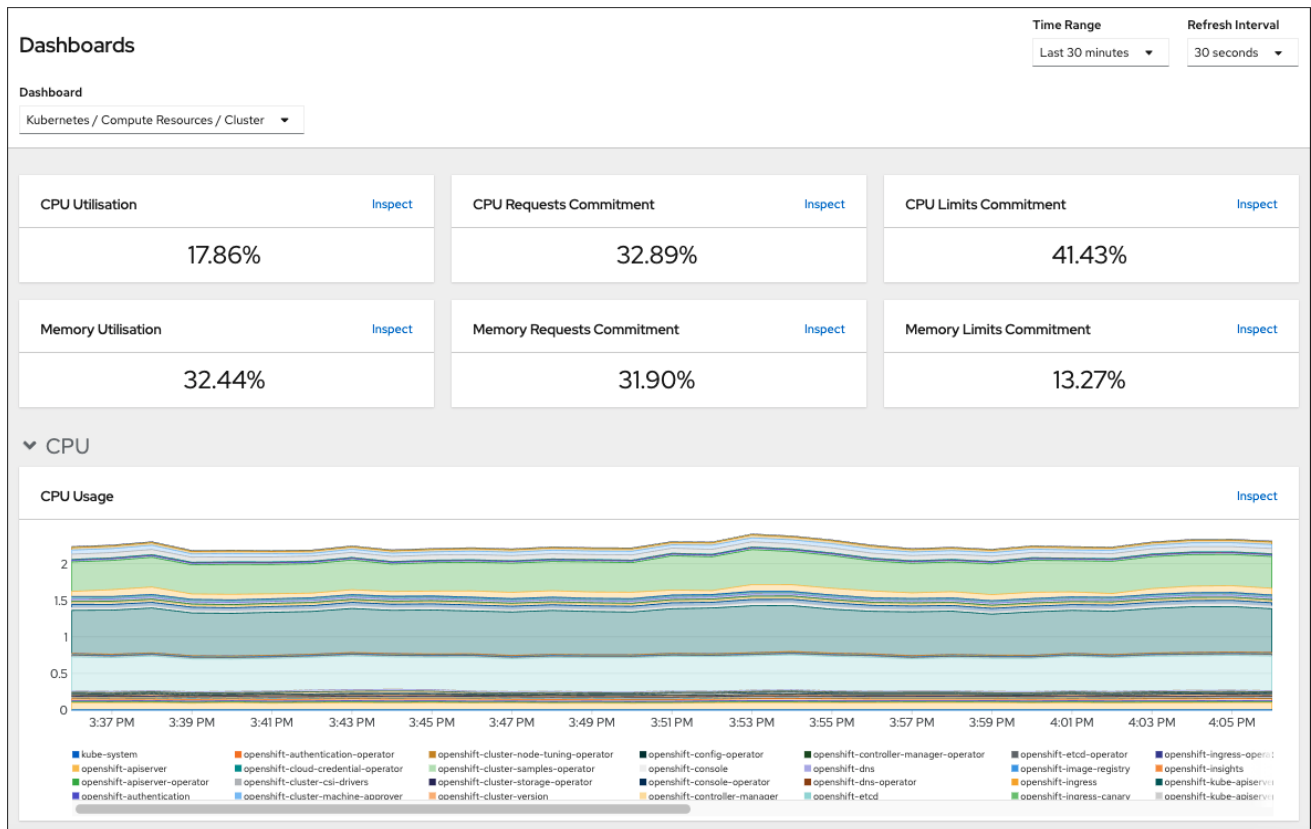
27.7. CHECKING CLUSTER RESOURCE AVAILABILITY AND UTILIZATION

OpenShift Container Platform provides a comprehensive set of monitoring dashboards that help you understand the state of cluster components.

In the **Administrator** perspective, you can access dashboards for core OpenShift Container Platform components, including:

- etcd
- Kubernetes compute resources
- Kubernetes network resources
- Prometheus
- Dashboards relating to cluster and node performance

Figure 27.1. Example compute resources dashboard



Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

- In the **Administrator** perspective in the OpenShift Container Platform web console, navigate to **Observe → Dashboards**.
- Choose a dashboard in the **Dashboard** list. Some dashboards, such as the **etcd** dashboard, produce additional sub-menus when selected.
- Optional: Select a time range for the graphs in the **Time Range** list.
 - Select a pre-defined time period.
 - Set a custom time range by selecting **Custom time range** in the **Time Range** list.
 - Input or select the **From** and **To** dates and times.
 - Click **Save** to save the custom time range.
- Optional: Select a **Refresh Interval**
- Hover over each of the graphs within a dashboard to display detailed information about specific items.

Additional resources

- See [Monitoring overview](#) for more information about the OpenShift Container Platform monitoring stack.

27.8. LISTING ALERTS THAT ARE FIRING

Alerts provide notifications when a set of defined conditions are true in an OpenShift Container Platform cluster. You can review the alerts that are firing in your cluster by using the Alerting UI in the OpenShift Container Platform web console.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. In the **Administrator** perspective, navigate to the **Observe → Alerting → Alerts** page.
2. Review the alerts that are firing, including their **Severity**, **State**, and **Source**.
3. Select an alert to view more detailed information in the **Alert Details** page.

Additional resources

- See [Managing alerts](#) for further details about alerting in OpenShift Container Platform.

27.9. NEXT STEPS

- See [Troubleshooting installations](#) if you experience issues when installing your cluster.
- After installing OpenShift Container Platform, you can [further expand and customize your cluster](#).

CHAPTER 28. TROUBLESHOOTING INSTALLATION ISSUES

To assist in troubleshooting a failed OpenShift Container Platform installation, you can gather logs from the bootstrap and control plane machines. You can also get debug information from the installation program. If you are unable to resolve the issue using the logs and debug information, see [Determining where installation issues occur](#) for component-specific troubleshooting.



NOTE

If your OpenShift Container Platform installation fails and the debug output or logs contain network timeouts or other connectivity errors, review the guidelines for [configuring your firewall](#). Gathering logs from your firewall and load balancer can help you diagnose network-related errors.

28.1. PREREQUISITES

- You attempted to install an OpenShift Container Platform cluster and the installation failed.

28.2. GATHERING LOGS FROM A FAILED INSTALLATION

If you gave an SSH key to your installation program, you can gather data about your failed installation.



NOTE

You use a different command to gather logs about an unsuccessful installation than to gather logs from a running cluster. If you must gather logs from a running cluster, use the **oc adm must-gather** command.

Prerequisites

- Your OpenShift Container Platform installation failed before the bootstrap process finished. The bootstrap node is running and accessible through SSH.
- The **ssh-agent** process is active on your computer, and you provided the same SSH key to both the **ssh-agent** process and the installation program.
- If you tried to install a cluster on infrastructure that you provisioned, you must have the fully qualified domain names of the bootstrap and control plane nodes.

Procedure

1. Generate the commands that are required to obtain the installation logs from the bootstrap and control plane machines:
 - If you used installer-provisioned infrastructure, change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install gather bootstrap --dir <installation_directory> 1
```

- 1** **installation_directory** is the directory you specified when you ran **./openshift-install create cluster**. This directory contains the OpenShift Container Platform definition files that the installation program creates.

For installer-provisioned infrastructure, the installation program stores information about the cluster, so you do not specify the hostnames or IP addresses.

- If you used infrastructure that you provisioned yourself, change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install gather bootstrap --dir <installation_directory> \ 1
--bootstrap <bootstrap_address> \ 2
--master <master_1_address> \ 3
--master <master_2_address> \ 4
--master <master_3_address>" 5
```

- 1** For **installation_directory**, specify the same directory you specified when you ran **./openshift-install create cluster**. This directory contains the OpenShift Container Platform definition files that the installation program creates.
- 2** **<bootstrap_address>** is the fully qualified domain name or IP address of the cluster's bootstrap machine.
- 3** **4** **5** For each control plane, or master, machine in your cluster, replace **<master_*_address>** with its fully qualified domain name or IP address.



NOTE

A default cluster contains three control plane machines. List all of your control plane machines as shown, no matter how many your cluster uses.

Example output

```
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "<installation_directory>/log-bundle-
<timestamp>.tar.gz"
```

If you open a Red Hat support case about your installation failure, include the compressed logs in the case.

28.3. MANUALLY GATHERING LOGS WITH SSH ACCESS TO YOUR HOST(S)

Manually gather logs in situations where **must-gather** or automated collection methods do not work.



IMPORTANT

By default, SSH access to the OpenShift Container Platform nodes is disabled on the Red Hat OpenStack Platform (RHOSP) based installations.

Prerequisites

- You must have SSH access to your host(s).

Procedure

1. Collect the **bootkube.service** service logs from the bootstrap host using the **journalctl** command by running:

```
$ journalctl -b -f -u bootkube.service
```

2. Collect the bootstrap host's container logs using the podman logs. This is shown as a loop to get all of the container logs from the host:

```
$ for pod in $(sudo podman ps -a -q); do sudo podman logs $pod; done
```

3. Alternatively, collect the host's container logs using the **tail** command by running:

```
# tail -f /var/lib/containers/storage/overlay-containers/*/userdata/ctr.log
```

4. Collect the **kubelet.service** and **crio.service** service logs from the master and worker hosts using the **journalctl** command by running:

```
$ journalctl -b -f -u kubelet.service -u crio.service
```

5. Collect the master and worker host container logs using the **tail** command by running:

```
$ sudo tail -f /var/log/containers/*
```

28.4. MANUALLY GATHERING LOGS WITHOUT SSH ACCESS TO YOUR HOST(S)

Manually gather logs in situations where **must-gather** or automated collection methods do not work.

If you do not have SSH access to your node, you can access the systems journal to investigate what is happening on your host.

Prerequisites

- Your OpenShift Container Platform installation must be complete.
- Your API service is still functional.
- You have system administrator privileges.

Procedure

1. Access **journald** unit logs under **/var/log** by running:

```
$ oc adm node-logs --role=master -u kubelet
```

2. Access host file paths under **/var/log** by running:

```
$ oc adm node-logs --role=master --path=openshift-apiserver
```

28.5. GETTING DEBUG INFORMATION FROM THE INSTALLATION PROGRAM

You can use any of the following actions to get debug information from the installation program.

- Look at debug messages from a past installation in the hidden **.openshift_install.log** file. For example, enter:

```
$ cat ~/<installation_directory>/.openshift_install.log 1
```

- 1 For **installation_directory**, specify the same directory you specified when you ran **./openshift-install create cluster**.

- Change to the directory that contains the installation program and re-run it with **--log-level=debug**:

```
$ ./openshift-install create cluster --dir <installation_directory> --log-level debug 1
```

- 1 For **installation_directory**, specify the same directory you specified when you ran **./openshift-install create cluster**.

28.6. REINSTALLING THE OPENSIFT CONTAINER PLATFORM CLUSTER

If you are unable to debug and resolve issues in the failed OpenShift Container Platform installation, consider installing a new OpenShift Container Platform cluster. Before starting the installation process again, you must complete thorough cleanup. For a user-provisioned infrastructure (UPI) installation, you must manually destroy the cluster and delete all associated resources. The following procedure is for an installer-provisioned infrastructure (IPI) installation.

Procedure

1. Destroy the cluster and remove all the resources associated with the cluster, including the hidden installer state files in the installation directory:

```
$ ./openshift-install destroy cluster --dir <installation_directory> 1
```

- 1 **installation_directory** is the directory you specified when you ran **./openshift-install create cluster**. This directory contains the OpenShift Container Platform definition files that the installation program creates.

2. Before reinstalling the cluster, delete the installation directory:

```
$ rm -rf <installation_directory>
```

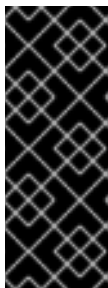
3. Follow the procedure for installing a new OpenShift Container Platform cluster.

Additional resources

- [Installing an OpenShift Container Platform cluster](#)

CHAPTER 29. SUPPORT FOR FIPS CRYPTOGRAPHY

You can install an OpenShift Container Platform cluster that uses FIPS validated or Modules In Process cryptographic libraries on the **x86_64**, **ppc64le**, and **s390x** architectures.



IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a RHEL 8 computer that is configured to operate in FIPS mode. Running RHEL 9 with FIPS mode enabled to install an OpenShift Container Platform cluster is not possible.

For more information about configuring FIPS mode on RHEL, see [Installing the system in FIPS mode](#).

For the Red Hat Enterprise Linux CoreOS (RHCOS) machines in your cluster, this change is applied when the machines are deployed based on the status of an option in the **install-config.yaml** file, which governs the cluster options that a user can change during cluster deployment. With Red Hat Enterprise Linux (RHEL) machines, you must enable FIPS mode when you install the operating system on the machines that you plan to use as worker machines. These configuration methods ensure that your cluster meets the requirements of a FIPS compliance audit: only FIPS validated or Modules In Process cryptography packages are enabled before the initial system boot.

Because FIPS must be enabled before the operating system that your cluster uses boots for the first time, you cannot enable FIPS after you deploy a cluster.

29.1. FIPS VALIDATION IN OPENSIFT CONTAINER PLATFORM

OpenShift Container Platform uses certain FIPS validated or Modules In Process modules within RHEL and RHCOS for the operating system components that it uses. See [RHEL8 core crypto components](#). For example, when users use SSH to connect to OpenShift Container Platform clusters and containers, those connections are properly encrypted.

OpenShift Container Platform components are written in Go and built with Red Hat's golang compiler. When you enable FIPS mode for your cluster, all OpenShift Container Platform components that require cryptographic signing call RHEL and RHCOS cryptographic libraries.

Table 29.1. FIPS mode attributes and limitations in OpenShift Container Platform 4.12

Attributes	Limitations
FIPS support in RHEL 8 and RHCOS operating systems.	The FIPS implementation does not offer a single function that both computes hash functions and validates the keys that are based on that hash. This limitation will continue to be evaluated and improved in future OpenShift Container Platform releases.
FIPS support in CRI-O runtimes.	
FIPS support in OpenShift Container Platform services.	
FIPS validated or Modules In Process cryptographic module and algorithms that are obtained from RHEL 8 and RHCOS binaries and images.	

Attributes	Limitations
Use of FIPS compatible golang compiler.	TLS FIPS support is not complete but is planned for future OpenShift Container Platform releases.
FIPS support across multiple architectures.	FIPS is currently only supported on OpenShift Container Platform deployments using the x86_64 , ppc64le , and s390x architectures.

29.2. FIPS SUPPORT IN COMPONENTS THAT THE CLUSTER USES

Although the OpenShift Container Platform cluster itself uses FIPS validated or Modules In Process modules, ensure that the systems that support your OpenShift Container Platform cluster use FIPS validated or Modules In Process modules for cryptography.

29.2.1. etcd

To ensure that the secrets that are stored in etcd use FIPS validated or Modules In Process encryption, boot the node in FIPS mode. After you install the cluster in FIPS mode, you can [encrypt the etcd data](#) by using the FIPS-approved **aes cbc** cryptographic algorithm.

29.2.2. Storage

For local storage, use RHEL-provided disk encryption or Container Native Storage that uses RHEL-provided disk encryption. By storing all data in volumes that use RHEL-provided disk encryption and enabling FIPS mode for your cluster, both data at rest and data in motion, or network data, are protected by FIPS validated or Modules In Process encryption. You can configure your cluster to encrypt the root filesystem of each node, as described in [Customizing nodes](#).

29.2.3. Runtimes

To ensure that containers know that they are running on a host that is using FIPS validated or Modules In Process cryptography modules, use CRI-O to manage your runtimes.

29.3. INSTALLING A CLUSTER IN FIPS MODE

To install a cluster in FIPS mode, follow the instructions to install a customized cluster on your preferred infrastructure. Ensure that you set **fips: true** in the **install-config.yaml** file before you deploy your cluster.

- [Amazon Web Services](#)
- [Alibaba Cloud](#)
- [Microsoft Azure](#)
- [Bare metal](#)
- [Google Cloud Platform](#)
- [IBM Cloud VPC](#)

- [IBM Power](#)
- [IBM Z and IBM® LinuxONE](#)
- [IBM Z and IBM® LinuxONE with RHEL KVM](#)
- [Red Hat OpenStack Platform \(RHOSP\)](#)
- [VMware vSphere](#)



NOTE

If you are using Azure File storage, you cannot enable FIPS mode.

To apply **AES CBC** encryption to your etcd data store, follow the [Encrypting etcd data](#) process after you install your cluster.

If you add RHEL nodes to your cluster, ensure that you enable FIPS mode on the machines before their initial boot. See [Adding RHEL compute machines to an OpenShift Container Platform cluster](#) and [Enabling FIPS Mode](#) in the RHEL 8 documentation.