OpenShift Container Platform 4.17

Installing on Azure Stack Hub

Installing OpenShift Container Platform on Azure Stack Hub

Last Updated: 2026-03-18

OpenShift Container Platform 4.17 Installing on Azure Stack Hub

Installing OpenShift Container Platform on Azure Stack Hub

Legal Notice

Abstract

This document describes how to install OpenShift Container Platform on Azure Stack Hub.

Table of Contents

CHAPTER 1. INSTALLATION METHODS

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See Installation process for more information about installer-provisioned and user-provisioned installation processes.

1.1. INSTALLING A CLUSTER ON INSTALLER-PROVISIONED INFRASTRUCTURE

You can install a cluster on Azure Stack Hub infrastructure that is provisioned by the OpenShift Container Platform installation program, by using the following method:

- **Installing a cluster**: You can install OpenShift Container Platform on Azure Stack Hub infrastructure that is provisioned by the OpenShift Container Platform installation program.

1.2. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE

You can install a cluster on Azure Stack Hub infrastructure that you provision, by using the following method:

- **Installing a cluster on Azure Stack Hub using ARM templates**You can install OpenShift Container Platform on Azure Stack Hub by using infrastructure that you provide. You can use the provided Azure Resource Manager (ARM) templates to assist with an installation.

1.3. ADDITIONAL RESOURCES

- Configuring an Azure Stack Hub account

CHAPTER 2. CONFIGURING AN AZURE STACK HUB ACCOUNT

Before you can install OpenShift Container Platform, you must configure a Microsoft Azure account.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

2.1. AZURE STACK HUB ACCOUNT LIMITS

The OpenShift Container Platform cluster uses a number of Microsoft Azure Stack Hub components, and the default Quota types in Azure Stack Hub affect your ability to install OpenShift Container Platform clusters.

The following table summarizes the Azure Stack Hub components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Description
vCPU	56	A default cluster requires 56 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances: • One bootstrap machine, which is removed after installation • Three control plane machines • Three compute machines Because the bootstrap, control plane, and worker machines use **Standard_DS4_v2** virtual machines, which use 8 vCPUs, a default cluster requires 56 vCPUs. The bootstrap node VM is used only during installation. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.
VNet	1	Each default cluster requires one Virtual Network (VNet), which contains two subnets.
Network interfaces	7	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.

<table>
<tr><th>Component</th><th>Number of components required by default</th><th>Description</th></tr>
<tr><td>Network security groups</td><td>2</td><td>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:<table><tr><td>controlplane</td><td>Allows the control plane machines to be reached on port 6443 from anywhere</td></tr><tr><td>node</td><td>Allows worker nodes to be reached from the internet on ports 80 and 443</td></tr></table></td></tr>
<tr><td>Network load balancers</td><td>3</td><td>Each cluster creates the following load balancers:<table><tr><td>default</td><td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td></tr><tr><td>internal</td><td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td></tr><tr><td>external</td><td>Public IP address that load balances requests to port 6443 across control plane machines</td></tr></table>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</td></tr>
<tr><td>Public IP addresses</td><td>2</td><td>The public load balancer uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</td></tr>
<tr><td>Private IP addresses</td><td>7</td><td>The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.</td></tr>
</table>

Additional resources

- Optimizing storage

2.2. CONFIGURING A DNS ZONE IN AZURE STACK HUB

To successfully install OpenShift Container Platform on Azure Stack Hub, you must create DNS records in an Azure Stack Hub DNS zone. The DNS zone must be authoritative for the domain. To delegate a registrar's DNS zone to Azure Stack Hub, see Microsoft's documentation for Azure Stack Hub

datacenter DNS integration.

2.3. REQUIRED AZURE STACK HUB ROLES

Your Microsoft Azure Stack Hub account must have the following roles for the subscription that you use:

- **Owner**

To set roles on the Azure portal, see the Manage access to resources in Azure Stack Hub with role-based access control in the Microsoft documentation.

2.4. CREATING A SERVICE PRINCIPAL

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the Azure CLI.
- Your Azure account has the required roles for the subscription that you use.

Procedure

1. Register your environment:

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

1 Specify the Azure Resource Manager endpoint, `https://management.<region>.<fqdn>/`.

   See the Microsoft documentation for details.

2. Set the active environment:

```
$ az cloud set -n AzureStackCloud
```

3. Update your environment configuration to use the specific API version for Azure Stack Hub:

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. Log in to the Azure CLI:

```
$ az login
```

   If you are in a multitenant environment, you must also supply the tenant ID.

5. If your Azure account uses subscriptions, ensure that you are using the right subscription:

   a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": AzureStackCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) Ensure that the value of the **tenantId** parameter is the correct subscription ID.

c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> (1)
```

(1) Specify the subscription ID.

d. Verify the subscription ID update:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

6. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.

7. Create the service principal for your account:

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ 1
  --scopes /subscriptions/<subscription_id> 2
  --years <years> 3
```

1. Specify the service principal name.
2. Specify the subscription ID.
3. Specify the number of years. By default, a service principal expires in one year. By using the **--years** option you can extend the validity of your service principal.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

8. Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

Additional resources

- About the Cloud Credential Operator

2.5. NEXT STEPS

- Install an OpenShift Container Platform cluster:

- Installing a cluster on Azure Stack Hub with customizations
- Install an OpenShift Container Platform cluster on Azure Stack Hub with user-provisioned infrastructure by following Installing a cluster on Azure Stack Hub using ARM templates .

CHAPTER 3. INSTALLER-PROVISIONED INFRASTRUCTURE

3.1. PREPARING TO INSTALL A CLUSTER ON AZURE STACK HUB

You prepare to install an OpenShift Container Platform cluster on Azure Stack Hub by completing the following steps:

- Verifying internet connectivity for your cluster.
- Configuring an Azure Stack Hub account .
- Generating an SSH key pair. You can use this key pair to authenticate into the OpenShift Container Platform cluster's nodes after it is deployed.
- Downloading the installation program.
- Installing the OpenShift CLI (**oc**).
- The Cloud Credential Operator (CCO) only supports your cloud provider in manual mode. As a result, you must manually manage clould credentials by specifying the identity and access management (IAM) secrets for your cloud provider.

3.1.1. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.17, you require access to the internet to install your cluster.

You must have internet access to perform the following actions:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

3.1.2. Generating a key pair for cluster node SSH access

To enable secure, passwordless SSH access to your cluster nodes, provide an SSH public key during the OpenShift Container Platform installation. This ensures that the installation program automatically configures the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for remote authentication through the **core** user.

The SSH public key gets added to the **~/.ssh/authorized_keys** list for the **core** user on each node. After the key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition

config files, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.

IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
```

   Specifies the path and file name, such as **~/.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your **~/.ssh** directory.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

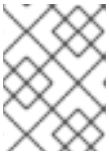
2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

   For example, run the following to view the **~/.ssh/id_ed25519.pub** public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.



   NOTE

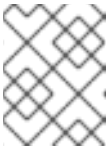
   On some distributions, default SSH private key identities such as **~/.ssh/id_rsa** and **~/.ssh/id_dsa** are managed automatically.

a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```



NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name>
```

Specifies the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

3.1.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

   TIP

   You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.

3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.

4. Place the downloaded file in the directory where you want to store the installation configuration files.

IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.
- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

3.1.4. Installing the OpenShift CLI on Linux

To manage your cluster and deploy applications from the command line, install the OpenShift CLI (**oc**) binary on Linux.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.17. Download and install the new version of **oc**.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.17 Linux Clients** entry and save the file.
5. Unpack the archive:

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

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

```
$ echo $PATH
```

Verification

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

```
$ oc <command>
```

3.1.5. Installing the OpenShift CLI on Windows

To manage your cluster and deploy applications from the command line, install OpenShift CLI (**oc**) binary on Windows.

IMPORTANT

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

Download and install the new version of **oc**.

Procedure

1. Navigate to the Download OpenShift Container Platform page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** list.
3. Click **Download Now** next to the **OpenShift v4.17 Windows Client** entry and save the file.
4. Extract the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH** variable.
   To check your **PATH** variable, open the command prompt and execute the following command:

```
C:\> path
```

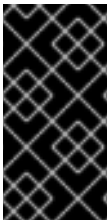
Verification

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

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

3.1.6. Installing the OpenShift CLI on macOS

To manage your cluster and deploy applications from the command line, install the OpenShift CLI (**oc**) binary on macOS.



IMPORTANT

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

Download and install the new version of **oc**.

Procedure

1. Navigate to the Download OpenShift Container Platform page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** list.
3. Select the appropriate version from the **Version** list.
4. Click **Download Now** next to the **OpenShift v4.17 macOS Clients** entry and save the file.

NOTE

For macOS arm64, choose the **OpenShift v4.17 macOS arm64 Client** entry.

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

```
$ echo $PATH
```

Verification

- Verify your installation by using an **oc** command:

```
$ oc <command>
```

3.1.7. Telemetry access for OpenShift Container Platform

To provide metrics about cluster health and the success of updates, the Telemetry service requires internet access. When connected, this service runs automatically by default and registers your cluster to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager,use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level. For more information about subscription watch, see "Data Gathered and Used by Red Hat's subscription services" in the *Additional resources* section.

Additional resources

- About remote health monitoring

3.2. INSTALLING A CLUSTER ON AZURE STACK HUB WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.17, you can install a cluster on Microsoft Azure Stack Hub with an installer-provisioned infrastructure. However, you must manually configure the **install-config.yaml** file to specify values that are specific to Azure Stack Hub.

NOTE

While you can select **azure** when using the installation program to deploy a cluster using installer-provisioned infrastructure, this option is only supported for the Azure Public Cloud.

3.2.1. Prerequisites

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You have installed Azure Stack Hub version 2008 or later.
- You configured an Azure Stack Hub account to host the cluster.
- If you use a firewall, you configured it to allow the sites that your cluster requires access to.
- You verified that you have approximately 16 GB of local disk space. Installing the cluster requires that you download the RHCOS virtual hard drive (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment. Decompressing the VHD files requires this amount of local disk space.

3.2.2. Uploading the RHCOS cluster image

You must download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment.

Prerequisites

- Generate the Ignition config files for your cluster.

Procedure

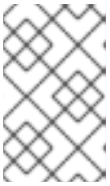
1. Obtain the RHCOS VHD cluster image:
   a. Export the URL of the RHCOS VHD to an environment variable.

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

   b. Download the compressed RHCOS VHD file locally.

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. Decompress the VHD file.



NOTE

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. The VHD file can be deleted once you have uploaded it.

3. Upload the local VHD to the Azure Stack Hub environment, making sure that the blob is publicly available. For example, you can upload the VHD to a blob using the **az** cli or the web portal.

3.2.3. Manually creating the installation configuration file

To customise your OpenShift Container Platform deployment and meet specific network requirements, manually create the installation configuration file. This ensures that the installation program uses your tailored settings rather than default values during the setup process.

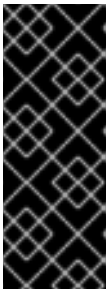
Prerequisites

- You have an SSH public key on your local machine for use with the installation program. You can use the key for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

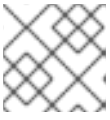
```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, such as bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the provided sample **install-config.yaml** file template and save the file in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

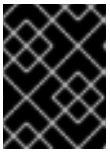
Make the following modifications:

a. Specify the required installation parameters.

b. Update the **platform.azure** section to specify the parameters that are specific to Azure Stack Hub.

c. Optional: Update one or more of the default configuration parameters to customize the installation.
For more information about the parameters, see "Installation configuration parameters".

3. Back up the **install-config.yaml** file so that you can use it to install many clusters.



IMPORTANT

Back up the **install-config.yaml** file now, because the installation process consumes the file in the next step.

Additional resources

- Installation configuration parameters for Azure Stack Hub

3.2.3.1. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Manual
controlPlane: (2) (3)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (4)
        diskType: premium_LRS
  replicas: 3
compute: (5)
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 (6)
        diskType: premium_LRS
  replicas: 3
metadata:
  name: test-cluster (7) (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (9)
```

```
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint (10) (11)
    baseDomainResourceGroupName: resource_group (12) (13)
    region: azure_stack_local_region (14) (15)
    resourceGroupName: existing_resource_group (16)
    outboundType: Loadbalancer
    cloudName: AzureStackCloud (17)
    clusterOSimage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
azurestack.x86_64.vhd (18) (19)
pullSecret: '{"auths": ...}' (20) (21)
fips: false (22)
sshKey: ssh-ed25519 AAAA... (23)
additionalTrustBundle: | (24)
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
```

(1) (7) (10) (12) (14) (17) (18) (20) Required.

(2) (5) If you do not provide these parameters and values, the installation program provides the default value.

(3) The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

(4) (6) You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

(8) The name of the cluster.

(9) The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

(11) The Azure Resource Manager endpoint that your Azure Stack Hub operator provides.

(13) The name of the resource group that contains the DNS zone for your base domain.

(15) The name of your Azure Stack Hub local region.

(16) The name of an existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

(19) The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.

(21) The pull secret required to authenticate your cluster.

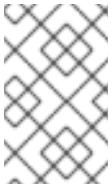
(22) Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography

modules that are provided with RHCOS instead.

IMPORTANT

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

23 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

24 If the Azure Stack Hub environment is using an internal Certificate Authority (CA), adding the CA certificate is required.

3.2.4. Manually manage cloud credentials

The Cloud Credential Operator (CCO) only supports your cloud provider in manual mode. As a result, you must specify the identity and access management (IAM) secrets for your cloud provider.

Procedure

1. If you have not previously created installation manifest files, do so by running the following command:

```
$ openshift-install create manifests --dir <installation_directory>
```

   where **<installation_directory>** is the directory in which the installation program creates files.

2. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. The **--included** parameter includes only the manifests that your specific cluster configuration requires.
2. Specify the location of the **install-config.yaml** file.
3. Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

4. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
```

```
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- Updating cloud provider resources with manually maintained credentials

3.2.5. Configuring the cluster to use an internal CA

If the Azure Stack Hub environment is using an internal Certificate Authority (CA), update the **cluster-proxy-01-config.yaml file** to configure the cluster to use the internal CA.

Prerequisites

- Create the **install-config.yaml** file and specify the certificate trust bundle in **.pem** format.
- Create the cluster manifests.

Procedure

1. From the directory in which the installation program creates files, go to the **manifests** directory.
2. Add **user-ca-bundle** to the **spec.trustedCA.name** field.

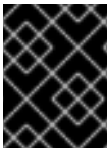
   Example cluster-proxy-01-config.yaml file

   ```
   apiVersion: config.openshift.io/v1
   kind: Proxy
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     trustedCA:
       name: user-ca-bundle
   status: {}
   ```

3. Optional: Back up the **manifests/ cluster-proxy-01-config.yaml** file. The installation program consumes the **manifests/** directory when you deploy the cluster.

3.2.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.



IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have verified that the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- In the directory that contains the installation program, initialize the cluster deployment by running the following command:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
```

```
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

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

3.2.7. Logging in to the cluster by using the CLI

To log in to your cluster as the default system user, export the **kubeconfig** file. This configuration enables the CLI to authenticate and connect to the specific API server created during OpenShift Container Platform installation.

The **kubeconfig** file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the OpenShift CLI (**oc**).

Procedure

1. Export the **kubeadmin** credentials by running the following command:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   where:

   <installation_directory>

   Specifies the path to the directory that stores the installation files.

2. Verify you can run **oc** commands successfully using the exported configuration by running the following command:

   ```
   $ oc whoami
   ```

   Example output

   ```
   -
   ```

```
system:admin
```

3.2.8. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

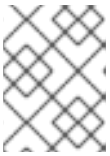
Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```



NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/.openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/.openshift_install.log** log file on the installation host.

Example output

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- Accessing the web console

3.2.9. Next steps

- Validating an installation
- Customize your cluster

- Optional: Remote health reporting
- Optional: Remove cloud provider credentials

3.3. INSTALLING A CLUSTER ON AZURE STACK HUB WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.17, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Azure Stack Hub. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

NOTE

While you can select **azure** when using the installation program to deploy a cluster using installer-provisioned infrastructure, this option is only supported for the Azure Public Cloud.

3.3.1. Prerequisites

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You have installed Azure Stack Hub version 2008 or later.
- You configured an Azure Stack Hub account to host the cluster.
- If you use a firewall, you configured it to allow the sites that your cluster requires access to.
- You verified that you have approximately 16 GB of local disk space. Installing the cluster requires that you download the RHCOS virtual hard drive (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment. Decompressing the VHD files requires this amount of local disk space.

3.3.2. Uploading the RHCOS cluster image

You must download the RHCOS virtual hard disk (VHD) cluster image and upload it to your Azure Stack Hub environment so that it is accessible during deployment.

Prerequisites

- Generate the Ignition config files for your cluster.

Procedure

1. Obtain the RHCOS VHD cluster image:
   a. Export the URL of the RHCOS VHD to an environment variable.

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

b. Download the compressed RHCOS VHD file locally.

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. Decompress the VHD file.

NOTE

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. The VHD file can be deleted once you have uploaded it.

3. Upload the local VHD to the Azure Stack Hub environment, making sure that the blob is publicly available. For example, you can upload the VHD to a blob using the **az** cli or the web portal.

3.3.3. Manually creating the installation configuration file

To customise your OpenShift Container Platform deployment and meet specific network requirements, manually create the installation configuration file. This ensures that the installation program uses your tailored settings rather than default values during the setup process.

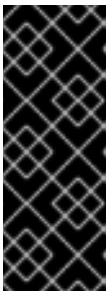
Prerequisites

- You have an SSH public key on your local machine for use with the installation program. You can use the key for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

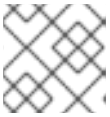
```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, such as bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the provided sample **install-config.yaml** file template and save the file in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

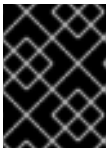
Make the following modifications:

a. Specify the required installation parameters.

b. Update the **platform.azure** section to specify the parameters that are specific to Azure Stack Hub.

c. Optional: Update one or more of the default configuration parameters to customize the installation.
For more information about the parameters, see "Installation configuration parameters".

3. Back up the **install-config.yaml** file so that you can use it to install many clusters.



IMPORTANT

Back up the **install-config.yaml** file now, because the installation process consumes the file in the next step.

Additional resources

- Installation configuration parameters for Azure Stack Hub

3.3.3.1. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Manual
controlPlane: (2) (3)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (4)
        diskType: premium_LRS
  replicas: 3
compute: (5)
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 (6)
        diskType: premium_LRS
  replicas: 3
metadata:
  name: test-cluster (7) (8)
```

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (9)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint (10) (11)
    baseDomainResourceGroupName: resource_group (12) (13)
    region: azure_stack_local_region (14) (15)
    resourceGroupName: existing_resource_group (16)
    outboundType: Loadbalancer
    cloudName: AzureStackCloud (17)
    clusterOSimage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
azurestack.x86_64.vhd (18) (19)
pullSecret: '{"auths": ...}' (20) (21)
fips: false (22)
sshKey: ssh-ed25519 AAAA... (23)
additionalTrustBundle: | (24)
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
```

(1) (7) (10) (12) (14) (17) (18) (20) Required.

(2) (5) If you do not provide these parameters and values, the installation program provides the default value.

(3) The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

(4) (6) You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

(8) The name of the cluster.

(9) The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

(11) The Azure Resource Manager endpoint that your Azure Stack Hub operator provides.

(13) The name of the resource group that contains the DNS zone for your base domain.

(15) The name of your Azure Stack Hub local region.

(16) The name of an existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

19 The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.

21 The pull secret required to authenticate your cluster.

22 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

23 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

24 If the Azure Stack Hub environment is using an internal Certificate Authority (CA), adding the CA certificate is required.

3.3.4. Manually manage cloud credentials

The Cloud Credential Operator (CCO) only supports your cloud provider in manual mode. As a result, you must specify the identity and access management (IAM) secrets for your cloud provider.

Procedure

1. If you have not previously created installation manifest files, do so by running the following command:

```
$ openshift-install create manifests --dir <installation_directory>
```

   where **<installation_directory>** is the directory in which the installation program creates files.

2. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
```

```
--credentials-requests \
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1. The **--included** parameter includes only the manifests that your specific cluster configuration requires.
2. Specify the location of the **install-config.yaml** file.
3. Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
    ...
```

4. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

-

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

Additional resources

- Updating a cluster using the web console
- Updating a cluster using the CLI

3.3.5. Configuring the cluster to use an internal CA

If the Azure Stack Hub environment is using an internal Certificate Authority (CA), update the **cluster-proxy-01-config.yaml file** to configure the cluster to use the internal CA.

Prerequisites

- Create the **install-config.yaml** file and specify the certificate trust bundle in **.pem** format.
- Create the cluster manifests.

Procedure

1. From the directory in which the installation program creates files, go to the **manifests** directory.
2. Add **user-ca-bundle** to the **spec.trustedCA.name** field.

Example cluster-proxy-01-config.yaml file

```
apiVersion: config.openshift.io/v1
kind: Proxy
metadata:
  creationTimestamp: null
  name: cluster
spec:
```

```
  trustedCA:
    name: user-ca-bundle
status: {}
```

3. Optional: Back up the **manifests/ cluster-proxy-01-config.yaml** file. The installation program consumes the **manifests/** directory when you deploy the cluster.

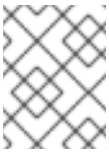
3.3.6. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  For more information, see "Installation configuration parameters".



NOTE

Set the **networking.machineNetwork** to match the Classless Inter-Domain Routing (CIDR) where the preferred subnet is located.

IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by **libVirt**. You cannot use any other CIDR range that overlaps with the **172.17.0.0/16** CIDR range for networks in your cluster.

Phase 2

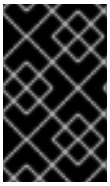
After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify an advanced network configuration.

During phase 2, you cannot override the values that you specified in phase 1 in the **install-config.yaml** file. However, you can customize the network plugin during phase 2.

3.3.7. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment.

You can specify advanced network configuration only before you install the cluster.



IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following example:

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

5. Remove the Kubernetes manifest files that define the control plane machines and compute **MachineSets**:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the **MachineSet** files to create compute machines by using the machine API, but you must update references to them to match your environment.

3.3.8. Cluster Network Operator configuration

To manage cluster networking, configure the Cluster Network Operator (CNO) **Network** custom resource (CR) named **cluster** so the cluster uses the correct IP ranges and network plugin settings for reliable pod and service connectivity. Some settings and fields are inherited at the time of install or by the **default.Network.type** plugin, OVN-Kubernetes.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin. **OVNKubernetes** is the only supported plugin during installation.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

3.3.8.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 3.1. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	**string**	The name of the CNO object. This name is always **cluster**.
spec.clusterNetwork	**array**	A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example: `spec:` `  clusterNetwork:` `  - cidr: 10.128.0.0/19` `  hostPrefix: 23` `  - cidr: 10.128.32.0/19` `  hostPrefix: 23`

Field	Type	Description
spec.serviceNetwork	**array**	A block of IP addresses for services. The OVN-Kubernetes network plugin supports only a single IP address block for the service network. For example: `spec:` `  serviceNetwork:` `  - 172.30.0.0/14` You can customize this field only in the **install-config.yaml** file before you create the manifests. The value is read-only in the manifest file.
spec.defaultNetwork	**object**	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	**object**	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

IMPORTANT

For a cluster that needs to deploy objects across multiple networks, ensure that you specify the same value for the **clusterNetwork.hostPrefix** parameter for each network type that is defined in the **install-config.yaml** file. Setting a different value for each **clusterNetwork.hostPrefix** parameter can impact the OVN-Kubernetes network plugin, where the plugin cannot effectively route object traffic among different nodes.

3.3.8.2. defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 3.2. **defaultNetwork** object

Field	Type	Description
type	**string**	**OVNKubernetes**. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation.  NOTE OpenShift Container Platform uses the OVN-Kubernetes network plugin by default. OpenShift SDN is no longer available as an installation choice for new clusters.

Field	Type	Description
ovnKubernetesConfig	**object**	This object is only valid for the OVN-Kubernetes network plugin.

3.3.8.3. Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 3.3. ovnKubernetesConfig object

Field	Type	Description
mtu	**integer**	The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to **100** less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of **9001**, and some have an MTU of**1500**, you must set this value to **1400**.
genevePort	**integer**	The port to use for all Geneve packets. The default value is **6081**. This value cannot be changed after cluster installation.
ipsecConfig	**object**	Specify a configuration object for customizing the IPsec configuration.
ipv4	**object**	Specifies a configuration object for IPv4 settings.
ipv6	**object**	Specifies a configuration object for IPv6 settings.
policyAuditConfig	**object**	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.

Field	Type	Description
gatewayConfig	**object**	Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway. Valid values are **Shared** and **Local**. The default value is **Shared**. In the default setting, the Open vSwitch (OVS) outputs traffic directly to the node IP interface. In the **Local** setting, it traverses the host network; consequently, it gets applied to the routing table of the host.  NOTE While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.

Table 3.4. **ovnKubernetesConfig.ipv4** object

Field	Type	Description
internalTransitSwitchSubnet	string	If your existing network infrastructure overlaps with the **100.88.0.0/16** IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. The subnet for the distributed transit switch that enables east-west traffic. This subnet cannot overlap with any other subnets used by OVN-Kubernetes or on the host itself. It must be large enough to accommodate one IP address per node in your cluster. The default value is **100.88.0.0/16**.
internalJoinSubnet	string	If your existing network infrastructure overlaps with the **100.64.0.0/16** IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the **clusterNetwork.cidr** value is **10.128.0.0/14** and the **clusterNetwork.hostPrefix** value is **/23**, then the maximum number of nodes is **2^(23-14)=512**. The default value is **100.64.0.0/16**.

Table 3.5. **ovnKubernetesConfig.ipv6** object

Field	Type	Description

Field	Type	Description
internalTransitSwitchSubnet	string	If your existing network infrastructure overlaps with the **fd97::/64** IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. The subnet for the distributed transit switch that enables east-west traffic. This subnet cannot overlap with any other subnets used by OVN-Kubernetes or on the host itself. It must be large enough to accommodate one IP address per node in your cluster. The default value is **fd97::/64**.
internalJoinSubnet	string	If your existing network infrastructure overlaps with the **fd98::/64** IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. The default value is **fd98::/64**.

Table 3.6. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is **20** messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is **50000000** or 50 MB.
maxLogFiles	integer	The maximum number of log files that are retained.
destination	string	One of the following additional audit log targets: **libc** The libc **syslog()** function of the journald process on the host. **udp:<host>:<port>** A syslog server. Replace **<host>:<port>** with the host and port of the syslog server. **unix:<file>** A Unix Domain Socket file specified by **<file>**. **null** Do not send the audit logs to any additional target.
syslogFacility	string	The syslog facility, such as **kern**, as defined by RFC5424. The default value is **local0**.

Table 3.7. **gatewayConfig** object

Field	Type	Description
routingViaHost	**boolean**	Set this field to **true** to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is **false**. This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to **true**, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.
ipForwarding	**object**	You can control IP forwarding for all traffic on OVN-Kubernetes managed interfaces by using the **ipForwarding** specification in the **Network** resource. Specify **Restricted** to only allow IP forwarding for Kubernetes related traffic. Specify **Global** to allow forwarding of all IP traffic. For new installations, the default is **Restricted**. For updates to OpenShift Container Platform 4.14 or later, the default is **Global**.  NOTE The default value of **Restricted** sets the IP forwarding to drop.
ipv4	**object**	Optional: Specify an object to configure the internal OVN-Kubernetes masquerade address for host to service traffic for IPv4 addresses.
ipv6	**object**	Optional: Specify an object to configure the internal OVN-Kubernetes masquerade address for host to service traffic for IPv6 addresses.

Table 3.8. **gatewayConfig.ipv4** object

Field	Type	Description

Field	Type	Description
internalMasqueradeSubnet	**string**	The masquerade IPv4 addresses that are used internally to enable host to service traffic. The host is configured with these IP addresses as well as the shared gateway bridge interface. The default value is **169.254.169.0/29**.  **IMPORTANT** For OpenShift Container Platform 4.17 and later versions, clusters use **169.254.0.0/17** as the default masquerade subnet. For upgraded clusters, there is no change to the default masquerade subnet.

Table 3.9. **gatewayConfig.ipv6** object

Field	Type	Description
internalMasqueradeSubnet	**string**	The masquerade IPv6 addresses that are used internally to enable host to service traffic. The host is configured with these IP addresses as well as the shared gateway bridge interface. The default value is **fd69::/125**.  **IMPORTANT** For OpenShift Container Platform 4.17 and later versions, clusters use **fd69::/112** as the default masquerade subnet. For upgraded clusters, there is no change to the default masquerade subnet.

Table 3.10. **ipsecConfig** object

Field	Type	Description
mode	**string**	Specifies the behavior of the IPsec implementation. Must be one of the following values: - **Disabled**: IPsec is not enabled on cluster nodes. - **External**: IPsec is enabled for network traffic with external hosts. - **Full**: IPsec is enabled for pod traffic and network traffic with external hosts.

Example OVN-Kubernetes configuration with IPSec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig:
      mode: Full
```

3.3.9. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with the OVN-Kubernetes network plugin. This allows a hybrid cluster that supports different node networking configurations.

NOTE

This configuration is necessary to run both Linux and Windows nodes in the same cluster.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   <installation_directory>

   Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

   ```
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```

   where:

   <installation_directory>

   Specifies the directory name that contains the **manifests/** directory for your cluster.

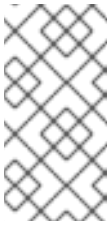
3. Open the **cluster-network-03-config.yml** file in an editor and configure OVN-Kubernetes with hybrid networking, as in the following example:

Specify a hybrid networking configuration

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
          hostPrefix: 23
        hybridOverlayVXLANPort: 9898 2
```

1 Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR must not overlap with the **clusterNetwork** CIDR.

2 Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **6081** port. For more information on this requirement, see Pod-to-pod connectivity between hosts is broken in the Microsoft documentation.



NOTE

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.

5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.

NOTE

For more information about using Linux and Windows nodes in the same cluster, see Understanding Windows container workloads.

3.3.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have verified that the cloud provider account on your host has the correct permissions to deploy the cluster. An account with incorrect permissions causes the installation process to fail with an error message that displays the missing permissions.

Procedure

- In the directory that contains the installation program, initialize the cluster deployment by running the following command:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

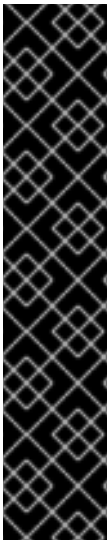
- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



IMPORTANT

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

3.3.11. Logging in to the cluster by using the CLI

To log in to your cluster as the default system user, export the **kubeconfig** file. This configuration enables the CLI to authenticate and connect to the specific API server created during OpenShift Container Platform installation.

The **kubeconfig** file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the OpenShift CLI (**oc**).

Procedure

1. Export the **kubeadmin** credentials by running the following command:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

   where:

   <installation_directory>

   Specifies the path to the directory that stores the installation files.

2. Verify you can run **oc** commands successfully using the exported configuration by running the following command:

```
$ oc whoami
```

   Example output

```
system:admin
```

3.3.12. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

NOTE

Alternatively, you can obtain the **kubeadmin** password from the **<installation_directory>/.openshift_install.log** log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the **<installation_directory>/.openshift_install.log** log file on the installation host.

Example output

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- Accessing the web console

3.3.13. Next steps

- Validating an installation
- Customize your cluster
- Optional: Remote health reporting
- Optional: Remove cloud provider credentials

CHAPTER 4. USER-PROVISIONED INFRASTRUCTURE

4.1. PREPARING TO INSTALL A CLUSTER ON AZURE STACK HUB

You prepare to install an OpenShift Container Platform cluster on Azure Stack Hub by completing the following steps:

- Verifying internet connectivity for your cluster.
- Configuring an Azure Stack Hub account .
- Generating an SSH key pair. You can use this key pair to authenticate into the OpenShift Container Platform cluster's nodes after it is deployed.
- Downloading the installation program.
- Installing the OpenShift CLI (**oc**).

4.1.1. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.17, you require access to the internet to install your cluster.

You must have internet access to perform the following actions:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

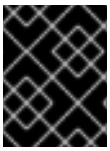
If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

4.1.2. Generating a key pair for cluster node SSH access

To enable secure, passwordless SSH access to your cluster nodes, provide an SSH public key during the OpenShift Container Platform installation. This ensures that the installation program automatically configures the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for remote authentication through the **core** user.

The SSH public key gets added to the ~/**.ssh**/**authorized_keys** list for the **core** user on each node. After the key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.



IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

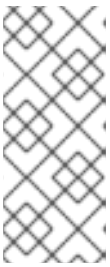
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
```

Specifies the path and file name, such as ~/**.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/**.ssh** directory.



NOTE

If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the ~/**.ssh/id_ed25519.pub** public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

NOTE

On some distributions, default SSH private key identities such as ~/**.ssh/id_rsa** and ~/**.ssh/id_dsa** are managed automatically.

a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name>
```

Specifies the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

4.1.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

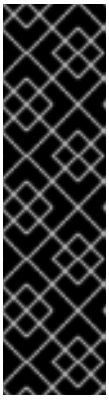
Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

TIP

You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.
3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.
4. Place the downloaded file in the directory where you want to store the installation configuration files.



IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.
- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

4.1.4. Installing the OpenShift CLI on Linux

To manage your cluster and deploy applications from the command line, install the OpenShift CLI (**oc**) binary on Linux.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.17. Download and install the new version of **oc**.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.17 Linux Clients** entry and save the file.
5. Unpack the archive:

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

6. Place the **oc** binary in a directory that is on your **PATH**.

To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

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

```
$ oc <command>
```

4.1.5. Installing the OpenShift CLI on Windows

To manage your cluster and deploy applications from the command line, install OpenShift CLI (**oc**) binary on Windows.

IMPORTANT

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

Download and install the new version of **oc**.

Procedure

1. Navigate to the Download OpenShift Container Platform page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** list.
3. Click **Download Now** next to the **OpenShift v4.17 Windows Client** entry and save the file.
4. Extract the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH** variable.
   To check your **PATH** variable, open the command prompt and execute the following command:

```
C:\> path
```

Verification

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

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

4.1.6. Installing the OpenShift CLI on macOS

To manage your cluster and deploy applications from the command line, install the OpenShift CLI (**oc**) binary on macOS.

IMPORTANT

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

Download and install the new version of **oc**.

Procedure

1. Navigate to the Download OpenShift Container Platform page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** list.
3. Select the appropriate version from the **Version** list.
4. Click **Download Now** next to the **OpenShift v4.17 macOS Clients** entry and save the file.

NOTE

For macOS arm64, choose the **OpenShift v4.17 macOS arm64 Client** entry.

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

```
$ echo $PATH
```

Verification

- Verify your installation by using an **oc** command:

```
$ oc <command>
```

4.2. INSTALLING A CLUSTER ON AZURE STACK HUB USING ARM TEMPLATES

In OpenShift Container Platform version 4.17, you can install a cluster on Microsoft Azure Stack Hub by using infrastructure that you provide.

Several Azure Resource Manager (ARM) templates are provided to assist in completing these steps or to help model your own.

IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several ARM templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

4.2.1. Prerequisites

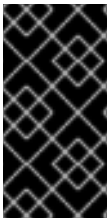
- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You have installed Azure Stack Hub version 2008 or later.
- You configured an Azure Stack Hub account to host the cluster.
- You downloaded the Azure CLI and installed it on your computer. See Install the Azure CLI in the Azure documentation. The documentation below was tested using version **2.28.0** of the Azure CLI. Azure CLI commands might perform differently based on the version you use.
- If you use a firewall and plan to use the Telemetry service, you configured the firewall to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

4.2.2. Configuring your Azure Stack Hub project

Before you can install OpenShift Container Platform, you must configure an Azure project to host it.



IMPORTANT

All Azure Stack Hub resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure Stack Hub restricts, see Resolve reserved resource name errors in the Azure documentation.

4.2.2.1. Azure Stack Hub account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure Stack Hub components, and the default Quota types in Azure Stack Hub affect your ability to install OpenShift Container Platform clusters.

The following table summarizes the Azure Stack Hub components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Description

<table>
<tr><th>Component</th><th>Number of components required by default</th><th>Description</th></tr>
<tr><td>vCPU</td><td>56</td><td>A default cluster requires 56 vCPUs, so you must increase the account limit.

By default, each cluster creates the following instances:
One bootstrap machine, which is removed after installation
Three control plane machines
Three compute machines
Because the bootstrap, control plane, and worker machines use Standard_DS4_v2 virtual machines, which use 8 vCPUs, a default cluster requires 56 vCPUs. The bootstrap node VM is used only during installation.

To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.</td></tr>
<tr><td>VNet</td><td>1</td><td>Each default cluster requires one Virtual Network (VNet), which contains two subnets.</td></tr>
<tr><td>Network interfaces</td><td>7</td><td>Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.</td></tr>
<tr><td>Network security groups</td><td>2</td><td>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:<table><tr><td>controlplane</td><td>Allows the control plane machines to be reached on port 6443 from anywhere</td></tr><tr><td>node</td><td>Allows worker nodes to be reached from the internet on ports 80 and 443</td></tr></table></td></tr>
</table>

<table>
<tr><th>Component</th><th>Number of components required by default</th><th>Description</th></tr>
<tr><td>Network load balancers</td><td>3</td><td>Each cluster creates the following load balancers:<table><tr><td>default</td><td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td></tr><tr><td>internal</td><td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td></tr><tr><td>external</td><td>Public IP address that load balances requests to port 6443 across control plane machines</td></tr></table>If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.</td></tr>
<tr><td>Public IP addresses</td><td>2</td><td>The public load balancer uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</td></tr>
<tr><td>Private IP addresses</td><td>7</td><td>The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.</td></tr>
</table>

Additional resources

- Optimizing storage

4.2.2.2. Configuring a DNS zone in Azure Stack Hub

To successfully install OpenShift Container Platform on Azure Stack Hub, you must create DNS records in an Azure Stack Hub DNS zone. The DNS zone must be authoritative for the domain. To delegate a registrar's DNS zone to Azure Stack Hub, see Microsoft's documentation for Azure Stack Hub datacenter DNS integration.

You can view Azure's DNS solution by visiting this example for creating DNS zones .

4.2.2.3. Certificate signing requests management

On user-provisioned infrastructure, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation when your cluster has limited access to automatic machine management.

The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

4.2.2.4. Required Azure Stack Hub roles

Your Microsoft Azure Stack Hub account must have the following roles for the subscription that you use:

- **Owner**

To set roles on the Azure portal, see the Manage access to resources in Azure Stack Hub with role-based access control in the Microsoft documentation.

4.2.2.5. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the Azure CLI.
- Your Azure account has the required roles for the subscription that you use.

Procedure

1. Register your environment:

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

1 Specify the Azure Resource Manager endpoint, `https://management.<region>.<fqdn>/`.

   See the Microsoft documentation for details.

2. Set the active environment:

```
$ az cloud set -n AzureStackCloud
```

3. Update your environment configuration to use the specific API version for Azure Stack Hub:

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. Log in to the Azure CLI:

```
$ az login
```

   If you are in a multitenant environment, you must also supply the tenant ID.

5. If your Azure account uses subscriptions, ensure that you are using the right subscription:

   a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

```
$ az account list --refresh
```

   Example output

```
[
  {
    "cloudName": AzureStackCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) Ensure that the value of the **tenantId** parameter is the correct subscription ID.

c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <subscription_id> (1)
```

(1) Specify the subscription ID.

d. Verify the subscription ID update:

```
$ az account show
```

Example output

```
{
  "environmentName": AzureStackCloud",
```

```
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

6. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.

7. Create the service principal for your account:

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ 1
  --scopes /subscriptions/<subscription_id> 2
  --years <years> 3
```

1. Specify the service principal name.
2. Specify the subscription ID.
3. Specify the number of years. By default, a service principal expires in one year. By using the **--years** option you can extend the validity of your service principal.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

8. Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

Additional resources

- About the Cloud Credential Operator

4.2.3. Creating the installation files for Azure Stack Hub

To install OpenShift Container Platform on Microsoft Azure Stack Hub using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You manually create the

install-config.yaml file, and then generate and customize the Kubernetes manifests and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

4.2.3.1. Manually creating the installation configuration file

To customise your OpenShift Container Platform deployment and meet specific network requirements, manually create the installation configuration file. This ensures that the installation program uses your tailored settings rather than default values during the setup process.

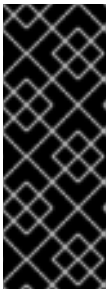
Prerequisites

- You have an SSH public key on your local machine for use with the installation program. You can use the key for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

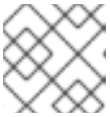
```
$ mkdir <installation_directory>
```



IMPORTANT

You must create a directory. Some installation assets, such as bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the provided sample **install-config.yaml** file template and save the file in the **<installation_directory>**.



NOTE

You must name this configuration file **install-config.yaml**.

Make the following modifications for Azure Stack Hub:

a. Set the **replicas** parameter to **0** for the **compute** pool:

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0
```

- **replicas**: Set to **0**.
  The compute machines will be provisioned manually later.

b. Update the **platform.azure** section of the **install-config.yaml** file to configure your Azure Stack Hub configuration:

```
platform:
  azure:
    armEndpoint: <azurestack_arm_endpoint>
    baseDomainResourceGroupName: <resource_group>
    cloudName: AzureStackCloud
    region: <azurestack_region>
```

where:

<azurestack_arm_endpoint>

Specifies the Azure Resource Manager endpoint of your Azure Stack Hub environment, like **https://management.local.azurestack.external**.

<resource_group>

Specifies the name of the resource group that contains the DNS zone for your base domain.

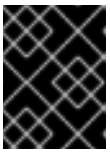
cloudName

Specifies the Azure Stack Hub environment, which is used to configure the Azure SDK with the appropriate Azure API endpoints.

region

Specifies the name of your Azure Stack Hub region.

3. Back up the **install-config.yaml** file so that you can use it to install many clusters.



IMPORTANT

Back up the **install-config.yaml** file now, because the installation process consumes the file in the next step.

Additional resources

- Installation configuration parameters for Azure Stack Hub

4.2.3.2. Sample customized install-config.yaml file for Azure Stack Hub

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. Use it as a resource to enter parameter values into the installation configuration file that you created manually.

```
apiVersion: v1
baseDomain: example.com
controlPlane: 1
  name: master
  platform:
    azure:
```

```
      osDisk:
        diskSizeGB: 1024 2
        diskType: premium_LRS
  replicas: 3
compute: 3
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 4
        diskType: premium_LRS
  replicas: 0
metadata:
  name: test-cluster 5
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 6
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint 7
    baseDomainResourceGroupName: resource_group 8
    region: azure_stack_local_region 9
    resourceGroupName: existing_resource_group 10
    outboundType: Loadbalancer
    cloudName: AzureStackCloud 11
pullSecret: '{"auths": ...}' 12
fips: false 13
additionalTrustBundle: | 14
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
sshKey: ssh-ed25519 AAAA... 15
```

1 3 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

2 4 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

5 Specify the name of the cluster.

6 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

7 Specify the Azure Resource Manager endpoint that your Azure Stack Hub operator provides.

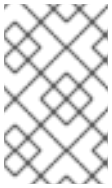
8. Specify the name of the resource group that contains the DNS zone for your base domain.
9. Specify the name of your Azure Stack Hub local region.
10. Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.
11. Specify the Azure Stack Hub environment as your target platform.
12. Specify the pull secret required to authenticate your cluster.
13. Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Installing the system in FIPS mode.

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

14. If your Azure Stack Hub environment uses an internal certificate authority (CA), add the necessary certificate bundle in **.pem** format.
15. You can optionally provide the **sshKey** value that you use to access the machines in your cluster.



NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

4.2.3.3. Configuring the cluster-wide proxy during installation

To enable internet access in environments that deny direct connections, configure a cluster-wide proxy in the **install-config.yaml** file. This configuration ensures that the new OpenShift Container Platform cluster routes traffic through the specified HTTP or HTTPS proxy.

Prerequisites

- You have an existing **install-config.yaml** file.
- You have reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to

bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud, Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
  httpsProxy: https://<username>:<pswd>@<ip>:<port>
  noProxy: example.com
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle>
# ...
```

where:

proxy.httpProxy

Specifies a proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

proxy.httpsProxy

Specifies a proxy URL to use for creating HTTPS connections outside the cluster.

proxy.noProxy

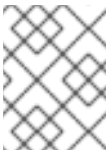
Specifies a comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.

additionalTrustBundle

If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace to hold the additional CA certificates. If you provide **additionalTrustBundle** and at least one proxy setting, the **Proxy** object is configured to reference the **user-ca-bundle** config map in the **trustedCA** field. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges the contents specified for the **trustedCA** parameter with the RHCOS trust bundle. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

additionalTrustBundlePolicy

Specifies the policy that determines the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**. Optional parameter.



NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

+

```
$ ./openshift-install wait-for install-complete --log-level debug
```

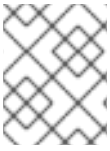
2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster** **Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

4.2.3.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure Stack Hub.



NOTE

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Export common variables found in the **install-config.yaml** to be used by the provided ARM templates:

```
$ export CLUSTER_NAME=<cluster_name>
```

- **<cluster_name>**: The value of the **.metadata.name** attribute from the **install-config.yaml** file.

```
$ export AZURE_REGION=<azure_region>
```

- **<azure_region>**: The region to deploy the cluster into. This is the value of the **.platform.azure.region** attribute from the **install-config.yaml** file.

```
$ export SSH_KEY=<ssh_key>
```

- **<ssh_key>**: The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the **.sshKey** attribute from the **install-config.yaml** file.

```
$ export BASE_DOMAIN=<base_domain>
```

- **<base_domain>**: The base domain to deploy the cluster to. The base domain corresponds to the DNS zone that you created for your cluster. This is the value of the **.baseDomain** attribute from the **install-config.yaml** file.

```
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group>
```

- **<base_domain_resource_group>**: The resource group where the DNS zone exists. This is the value of the **.platform.azure.baseDomainResourceGroupName** attribute from the **install-config.yaml** file.
  For example:

```
$ export CLUSTER_NAME=test-cluster
```

```
$ export AZURE_REGION=centralus
```

```
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
```

```
$ export BASE_DOMAIN=example.com
```

```
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. Export the kubeadmin credentials:

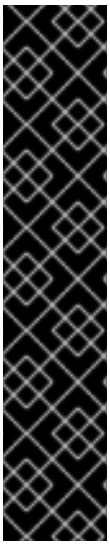
```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

- **<installation_directory>**: Specify the path to the directory that you stored the installation files in.

4.2.3.5. Creating the Kubernetes manifest and Ignition config files

To customize cluster definitions and manually start machines, generate the Kubernetes manifest and Ignition config files. These assets provide the necessary instructions to configure the cluster infrastructure according to your specific deployment requirements.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.



IMPORTANT

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

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

   where

   <installation_directory>

   Specifies the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

   By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the control plane machine set:

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. Remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

IMPORTANT

If you disabled the **MachineAPI** capability when installing a cluster on user-provisioned infrastructure, you must remove the Kubernetes manifest files that define the worker machines. Otherwise, your cluster fails to install.

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

5. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.

   b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.

   c. Save and exit the file.

6. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone:
    id: mycluster-100419-private-zone
  publicZone: 1
    id: example.openshift.com
status: {}
```

spec.privateZone: Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

7. Optional: If your Azure Stack Hub environment uses an internal certificate authority (CA), you must update the **.spec.trustedCA.name** field in the **<installation_directory>/manifests/cluster-proxy-01-config.yaml** file to use **user-ca-bundle**:

```
...
spec:
  trustedCA:
    name: user-ca-bundle
...
```

Later, you must update your bootstrap ignition to include the CA.

8. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

   a. Export the infrastructure ID by using the following command:

   ```
   $ export INFRA_ID=<infra_id>
   ```

   where:

   <infra_id>

   Specifies that the OpenShift Container Platform cluster has been assigned an identifier (**INFRA_ID**) in the form of **<cluster_name>-<random_string>**. This identifier is used as the base name for most resources created using the provided ARM templates. This is the value of the **.status.infrastructureName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

   b. Export the resource group by using the following command:

   ```
   $ export RESOURCE_GROUP=<resource_group>
   ```

   where:

   <resource_group>

   All resources created in this Azure deployment exists as part of a resource group. The resource group name is also based on the **INFRA_ID**, in the form of **<cluster_name>-<random_string>-rg**. This is the value of the **.status.platformStatus.azure.resourceGroupName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

9. Manually create your cloud credentials.

   a. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your **openshift-install** binary is built to use:

   ```
   $ openshift-install version
   ```

   Example output

   ```
   release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
   ```

   b. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   c. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \//
   ```

```
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml
\//
  --to=<path_to_directory_for_credentials_requests>
```

where:

--included

Specifies to include only the manifests that your specific cluster configuration requires.

<path_to_directory_with_installation_configuration>

Specifies the location of the **install-config.yaml** file.

<path_to_directory_for_credentials_requests>

Specifies the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.
This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-azure
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
```

d. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object. The format for the secret data varies for each cloud provider.

Sample secrets.yaml file

```
apiVersion: v1
kind: Secret
metadata:
    name: ${secret_name}
    namespace: ${secret_namespace}
stringData:
  azure_subscription_id: ${subscription_id}
  azure_client_id: ${app_id}
  azure_client_secret: ${client_secret}
  azure_tenant_id: ${tenant_id}
```

```
  azure_resource_prefix: ${cluster_name}
  azure_resourcegroup: ${resource_group}
  azure_region: ${azure_region}
```

e. Create a **cco-configmap.yaml** file in the manifests directory with the Cloud Credential Operator (CCO) disabled:

Sample ConfigMap object

```
apiVersion: v1
kind: ConfigMap
metadata:
name: cloud-credential-operator-config
namespace: openshift-cloud-credential-operator
  annotations:
    release.openshift.io/create-only: "true"
data:
  disabled: "true"
```

10. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

where:

<installation_directory>

Specifies the same installation directory.
Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├──── auth
│     ├──── kubeadmin-password
│     └──── kubeconfig
├──── bootstrap.ign
├──── master.ign
├──── metadata.json
└──── worker.ign
```

Additional resources

- Manually manage cloud credentials

4.2.3.6. Optional: Creating a separate /var partition

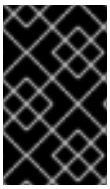
It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.



IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.17.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 1 The storage device name of the disk that you want to partition.

2 When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

3 The size of the data partition in mebibytes.

4 The **prjquota** mount option must be enabled for filesystems used for container storage.

NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
```

```
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

You can now use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

4.2.4. Creating the Azure resource group

You must create a Microsoft Azure resource group. This is used during the installation of your OpenShift Container Platform cluster on Azure Stack Hub.

Procedure

- Create the resource group in a supported Azure region:

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

4.2.5. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally. You must copy and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a storage container so they are accessible during deployment.

Prerequisites

- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

WARNING

The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your **CLUSTER_NAME** variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see Resolve errors for storage account names in the Azure documentation.

2. Export the storage account key as an environment variable:

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. Export the URL of the RHCOS VHD to an environment variable:

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r '.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Create the storage container for the VHD:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

5. Download the compressed RHCOS VHD file locally:

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

6. Decompress the VHD file.

NOTE

The decompressed VHD file is approximately 16 GB, so be sure that your host system has 16 GB of free space available. You can delete the VHD file after you upload it.

7. Copy the local VHD to a blob:

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -f rhcos-<rhcos_version>-azurestack.x86_64.vhd
```

8. Create a blob storage container and upload the generated **bootstrap.ign** file:

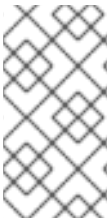
```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

4.2.6. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, Azure Stack Hub's datacenter DNS integration is used, so you will create a DNS zone.



NOTE

The DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the DNS zone; be sure the installation config you generated earlier reflects that scenario.

Procedure

- Create the new DNS zone in the resource group exported in the **BASE_DOMAIN_RESOURCE_GROUP** environment variable:

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

  You can skip this step if you are using a DNS zone that already exists.

You can learn more about configuring a DNS zone in Azure Stack Hub by visiting that section.

4.2.7. Creating a VNet in Azure Stack Hub

You must create a virtual network (VNet) in Microsoft Azure Stack Hub for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your Azure Stack Hub infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for the VNet**section of this topic and save it as **01_vnet.json** in your cluster's installation directory. This template describes the VNet that your cluster requires.

2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

4.2.7.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

Example 4.1. 01_vnet.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "addressPrefix" : "10.0.0.0/16",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetPrefix" : "10.0.0.0/24",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetPrefix" : "10.0.1.0/24",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/virtualNetworks",
      "name" : "[variables('virtualNetworkName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
      ],
      "properties" : {
        "addressSpace" : {
          "addressPrefixes" : [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets" : [
```

```
        {
          "name" : "[variables('masterSubnetName')]",
          "properties" : {
            "addressPrefix" : "[variables('masterSubnetPrefix')]",
            "serviceEndpoints": [],
            "networkSecurityGroup" : {
              "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
            }
          }
        },
        {
          "name" : "[variables('nodeSubnetName')]",
          "properties" : {
            "addressPrefix" : "[variables('nodeSubnetPrefix')]",
            "serviceEndpoints": [],
            "networkSecurityGroup" : {
              "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
            }
          }
        }
      ]
    }
  },
  {
    "type" : "Microsoft.Network/networkSecurityGroups",
    "name" : "[variables('clusterNsgName')]",
    "apiVersion" : "2017-10-01",
    "location" : "[variables('location')]",
    "properties" : {
      "securityRules" : [
        {
          "name" : "apiserver_in",
          "properties" : {
            "protocol" : "Tcp",
            "sourcePortRange" : "*",
            "destinationPortRange" : "6443",
            "sourceAddressPrefix" : "*",
            "destinationAddressPrefix" : "*",
            "access" : "Allow",
            "priority" : 101,
            "direction" : "Inbound"
          }
        },
        {
          "name" : "ign_in",
          "properties" : {
            "protocol" : "*",
            "sourcePortRange" : "*",
            "destinationPortRange" : "22623",
            "sourceAddressPrefix" : "*",
            "destinationAddressPrefix" : "*",
            "access" : "Allow",
            "priority" : 102,
            "direction" : "Inbound"
```

```
          }
        }
      ]
    }
  }
 ]
}
```

4.2.8. Deploying the RHCOS cluster image for the Azure Stack Hub infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure Stack Hub for your OpenShift Container Platform nodes.

Prerequisites

- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.
- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the **ARM template for image storage**section of this topic and save it as **02_storage.json** in your cluster's installation directory. This template describes the image storage that your cluster requires.
2. Export the RHCOS VHD blob URL as a variable:

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. Deploy the cluster image:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1. The blob URL of the RHCOS VHD to be used to create master and worker machines.
2. The base name to be used in resource names; this is usually the cluster's infrastructure ID.
3. The name of your Azure storage account.
4. Specify the system architecture. Valid values are **x64** (default) or **Arm64**.

4.2.8.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

-

Example 4.2. 02_storage.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vhdBlobURL" : {
      "type" : "string",
      "metadata" : {
        "description" : "URL pointing to the blob where the VHD to be used to create master and worker machines is located"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "imageName" : "[parameters('baseName')]"
  },
  "resources" : [
    {
      "apiVersion" : "2017-12-01",
      "type": "Microsoft.Compute/images",
      "name": "[variables('imageName')]",
      "location" : "[variables('location')]",
      "properties": {
        "storageProfile": {
          "osDisk": {
            "osType": "Linux",
            "osState": "Generalized",
            "blobUri": "[parameters('vhdBlobURL')]",
            "storageAccountType": "Standard_LRS"
          }
        }
      }
    }
  ]
}
```

4.2.9. Networking requirements for user-provisioned infrastructure

You must configure networking for all the Red Hat Enterprise Linux CoreOS (RHCOS) machines in **initramfs** during boot, so that they can fetch their Ignition config files.

IMPORTANT

Ensure you enable the **disk.EnableUUID** parameter on all virtual machines in your cluster.

4.2.9.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 4.1. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
	22623	The port handles traffic from the Machine Config Server and directs the traffic to the control plane machines.
UDP	**6081**	Geneve
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123**. If an external NTP time server is configured, you must open UDP port **123**.
	TCP/UDP	**30000**-**32767**

Protocol	Port	Description
Kubernetes node port	ESP	N/A

Table 4.2. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	**6443**	Kubernetes API

Table 4.3. Ports used for control plane machine to control plane machine communications

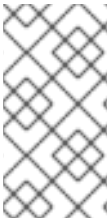
Protocol	Port	Description
TCP	**2379-2380**	etcd server and peer ports

4.2.10. Creating networking and load balancing components in Azure Stack Hub

You must configure networking and load balancing in Microsoft Azure Stack Hub for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.

Load balancing requires the following DNS records:

- An **api** DNS record for the API public load balancer in the DNS zone.
- An **api-int** DNS record for the API internal load balancer in the DNS zone.



NOTE

If you do not use the provided ARM template to create your Azure Stack Hub infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure a VNet and associated subnets in Azure Stack Hub.

Procedure

1. Copy the template from the **ARM template for the network and load balancers**section of this topic and save it as **03_infra.json** in your cluster's installation directory. This template describes the networking and load balancing objects that your cluster requires.
2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Create an **api** DNS record and an **api-int** DNS record. When creating the API DNS records, the **${BASE_DOMAIN_RESOURCE_GROUP}** variable must point to the resource group where the DNS zone exists.

   a. Export the following variable:

   ```
   $ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
   ```

   b. Export the following variable:

   ```
   $ export PRIVATE_IP=`az network lb frontend-ip show -g "$RESOURCE_GROUP" --lb-name "${INFRA_ID}-internal" -n internal-lb-ip --query "privateIpAddress" -o tsv`
   ```

   c. Create the **api** DNS record in a new DNS zone:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
   ```

   If you are adding the cluster to an existing DNS zone, you can create the **api** DNS record in it instead:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
   ```

   d. Create the **api-int** DNS record in a new DNS zone:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z "${CLUSTER_NAME}.${BASE_DOMAIN}" -n api-int -a ${PRIVATE_IP} --ttl 60
   ```

   If you are adding the cluster to an existing DNS zone, you can create the **api-int** DNS record in it instead:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n api-int.${CLUSTER_NAME} -a ${PRIVATE_IP} --ttl 60
   ```

4.2.10.1. ARM template for the network and load balancers

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

Example 4.3. 03_infra.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
```

```
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
    "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'))]",
    "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
    "masterAvailabilitySetName" : "[concat(parameters('baseName'), '-cluster')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal')]",
    "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
    "skuName": "Basic"
  },
  "resources" : [
    {
      "apiVersion": "2017-03-30",
      "type" : "Microsoft.Compute/availabilitySets",
      "name" : "[variables('masterAvailabilitySetName')]",
      "location" : "[variables('location')]",
      "properties": {
        "platformFaultDomainCount": "2",
        "platformUpdateDomainCount": "5"
      },
      "sku": {
        "name": "Aligned"
      }
    },
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('masterPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
```

```
          "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('masterLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
      ],
      "properties" : {
        "frontendIPConfigurations" : [
          {
            "name" : "public-lb-ip",
            "properties" : {
              "publicIPAddress" : {
                "id" : "[variables('masterPublicIpAddressID')]"
              }
            }
          }
        ],
        "backendAddressPools" : [
          {
            "name" : "[variables('masterLoadBalancerName')]"
          }
        ],
        "loadBalancingRules" : [
          {
            "name" : "api-public",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-
ip')]"
              },
              "backendAddressPool" : {
                "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
              },
              "protocol" : "Tcp",
              "loadDistribution" : "Default",
              "idleTimeoutInMinutes" : 30,
              "frontendPort" : 6443,
              "backendPort" : 6443,
              "probe" : {
                "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-public-probe')]"
              }
            }
          }
        ],
        "probes" : [
          {
```

```
          "name" : "api-public-probe",
          "properties" : {
            "protocol" : "Tcp",
            "port" : 6443,
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2017-10-01",
    "type" : "Microsoft.Network/loadBalancers",
    "name" : "[variables('internalLoadBalancerName')]",
    "location" : "[variables('location')]",
    "sku": {
      "name": "[variables('skuName')]"
    },
    "properties" : {
      "frontendIPConfigurations" : [
        {
          "name" : "internal-lb-ip",
          "properties" : {
            "privateIPAllocationMethod" : "Dynamic",
            "subnet" : {
              "id" : "[variables('masterSubnetRef')]"
            },
            "privateIPAddressVersion" : "IPv4"
          }
        }
      ],
      "backendAddressPools" : [
        {
          "name" : "[variables('internalLoadBalancerName')]"
        }
      ],
      "loadBalancingRules" : [
        {
          "name" : "api-internal",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
            },
            "frontendPort" : 6443,
            "backendPort" : 6443,
            "enableFloatingIP" : false,
            "idleTimeoutInMinutes" : 30,
            "protocol" : "Tcp",
            "enableTcpReset" : false,
            "loadDistribution" : "Default",
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/',
variables('internalLoadBalancerName'))]"
            },
```

```
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        },
        {
          "name" : "sint",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
            },
            "frontendPort" : 22623,
            "backendPort" : 22623,
            "enableFloatingIP" : false,
            "idleTimeoutInMinutes" : 30,
            "protocol" : "Tcp",
            "enableTcpReset" : false,
            "loadDistribution" : "Default",
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/',
variables('internalLoadBalancerName'))]"
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Tcp",
            "port" : 6443,
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        },
        {
          "name" : "sint-probe",
          "properties" : {
            "protocol" : "Tcp",
            "port" : 22623,
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
  }
 ]
}
```

4.2.11. Creating the bootstrap machine in Azure Stack Hub

You must create the bootstrap machine in Microsoft Azure Stack Hub to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure networking and load balancers in Azure Stack Hub.

Procedure

1. Copy the template from the **ARM template for the bootstrap machine**section of this topic and save it as **04_bootstrap.json** in your cluster's installation directory. This template describes the bootstrap machine that your cluster requires.
2. Export the bootstrap URL variable:

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. Export the bootstrap ignition variable:
   a. If your environment uses a public certificate authority (CA), run this command:

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

   b. If your environment uses an internal CA, you must add your PEM encoded bundle to the bootstrap ignition stub so that your bootstrap virtual machine can pull the bootstrap ignition from the storage account. Run the following commands, which assume your CA is in a file called **CA.pem**:

```
$ export CA="data:text/plain;charset=utf-8;base64,$(cat CA.pem |base64 |tr -d '\n')"
```

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url "$BOOTSTRAP_URL" --arg cert "$CA" '{ignition:{version:$v,security:{tls:{certificateAuthorities:[{source:$cert}]}},config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

4. Create the deployment by using the **az** CLI:

```
$ az deployment group create --verbose -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. The bootstrap Ignition content for the bootstrap cluster.
2. The base name to be used in resource names; this is usually the cluster's infrastructure ID.
3. The name of the storage account for your cluster.

4.2.11.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 4.4. 04_bootstrap.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string."
      }
    },
    "diagnosticsStorageAccountName": {
      "type": "string"
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_DS4_v2",
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    }
```

```
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'))]",
    "masterAvailabilitySetName" : "[concat(parameters('baseName'), '-cluster')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName" : "[concat(variables('vmName'), '-nic')]",
    "imageName" : "[parameters('baseName')]",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('sshPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "Basic"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "name" : "[variables('nicName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
      ],
      "properties" : {
        "securityRules": [
          {
            "properties": {
              "description": "ssh-in-nic",
              "protocol": "Tcp",
              "sourcePortRange": "*",
              "destinationPortRange": "22"
            }}],
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
```

```
            "privateIPAllocationMethod" : "Dynamic",
            "publicIPAddress": {
              "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
            },
            "subnet" : {
              "id" : "[variables('masterSubnetRef')]"
            },
            "loadBalancerBackendAddressPools" : [
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
              },
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/',
variables('internalLoadBalancerName'))]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "name": "[parameters('diagnosticsStorageAccountName')]",
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2017-10-01",
    "location": "[variables('location')]",
    "properties": {},
    "kind": "Storage",
    "sku": {
      "name": "Standard_LRS"
    }
  },
  {
    "apiVersion" : "2017-12-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "name" : "[variables('vmName')]",
    "location" : "[variables('location')]",
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]",
      "[concat('Microsoft.Storage/storageAccounts/',
parameters('diagnosticsStorageAccountName'))]"
    ],
    "properties" : {
      "availabilitySet": {
        "id": "
[resourceId('Microsoft.Compute/availabilitySets',variables('masterAvailabilitySetName'))]"
      },
      "hardwareProfile" : {
        "vmSize" : "[parameters('bootstrapVMSize')]"
      },
```

```
        "osProfile" : {
          "computerName" : "[variables('vmName')]",
          "adminUsername" : "core",
          "customData" : "[parameters('bootstrapIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : true,
            "ssh" : {
              "publicKeys" : [
                {
                  "path" : "[variables('sshKeyPath')]",
                  "keyData" : "[parameters('sshKeyData')]"
                }
              ]
            }
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmName'),'_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "managedDisk": {
              "storageAccountType": "Standard_LRS"
            },
            "diskSizeGB" : 100
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
            }
          ]
        },
        "diagnosticsProfile": {
          "bootDiagnostics": {
            "enabled": true,
            "storageUri": "[reference(resourceId('Microsoft.Storage/storageAccounts',
parameters('diagnosticsStorageAccountName'))).primaryEndpoints.blob]"
          }
        }
      }
    },
    {
      "apiVersion" : "2017-10-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
      ],
      "properties": {
        "protocol" : "Tcp",
```

```
            "sourcePortRange" : "*",
            "destinationPortRange" : "22",
            "sourceAddressPrefix" : "*",
            "destinationAddressPrefix" : "*",
            "access" : "Allow",
            "priority" : 100,
            "direction" : "Inbound"
          }
        }
      ]
    }
```

4.2.12. Creating the control plane machines in Azure Stack Hub

You must create the control plane machines in Microsoft Azure Stack Hub for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Create the bootstrap machine.

Procedure

1. Copy the template from the **ARM template for control plane machines**section of this topic and save it as **05_masters.json** in your cluster's installation directory. This template describes the control plane machines that your cluster requires.

2. Export the following variable needed by the control plane machine deployment:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. The Ignition content for the control plane nodes (also known as the master nodes).
2. The base name to be used in resource names; this is usually the cluster's infrastructure ID.
3. The name of the storage account for your cluster.

4.2.12.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 4.5. 05_masters.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "masterIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the master nodes"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string"
      }
    },
    "diagnosticsStorageAccountName": {
      "type": "string"
    },
    "masterVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_DS4_v2",
      "metadata" : {
        "description" : "The size of the Master Virtual Machines"
      }
    },
    "diskSizeGB" : {
      "type" : "int",
      "defaultValue" : 1023,
      "metadata" : {
        "description" : "Size of the Master VM OS disk, in GB"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks', variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/', variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[concat(parameters('baseName'))]",
```

```
    "masterAvailabilitySetName" : "[concat(parameters('baseName'), '-cluster')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]",
    "imageName" : "[parameters('baseName')]",
    "numberOfMasters" : 3,
    "vms" : {
      "copy" : [
        {
          "name" : "vmNames",
          "count" :  "[variables('numberOfMasters')]",
          "input" : {
            "name" : "[concat(parameters('baseName'), string('-master-'),
string(copyIndex('vmNames')))]"
          }
        }
      ]
    }
  },
  "resources" : [
    {
      "name": "[parameters('diagnosticsStorageAccountName')]",
      "type": "Microsoft.Storage/storageAccounts",
      "apiVersion": "2017-10-01",
      "location": "[variables('location')]",
      "properties": {},
      "kind": "Storage",
      "sku": {
        "name": "Standard_LRS"
      }
    },
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "location": "[variables('location')]",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[variables('numberOfMasters')]"
      },
      "name" : "[concat(variables('vms').vmNames[copyIndex()].name, '-nic')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
                },
```

```
            {
              "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/',
variables('internalLoadBalancerName'))]"
            }
          ]
        }
      }
    ]
  }
 },
 {
  "apiVersion" : "2017-12-01",
  "type" : "Microsoft.Compute/virtualMachines",
  "location" : "[variables('location')]",
  "copy" : {
   "name" : "vmCopy",
   "count" : "[variables('numberOfMasters')]"
  },
  "name" : "[variables('vms').vmNames[copyIndex()].name]",
  "dependsOn" : [
   "[concat('Microsoft.Network/networkInterfaces/',
concat(variables('vms').vmNames[copyIndex()].name, '-nic'))]",
   "[concat('Microsoft.Storage/storageAccounts/',
parameters('diagnosticsStorageAccountName'))]"
  ],
  "properties" : {
   "availabilitySet": {
    "id": "
[resourceId('Microsoft.Compute/availabilitySets',variables('masterAvailabilitySetName'))]"
   },
   "hardwareProfile" : {
    "vmSize" : "[parameters('masterVMSize')]"
   },
   "osProfile" : {
    "computerName" : "[variables('vms').vmNames[copyIndex()].name]",
    "adminUsername" : "core",
    "customData" : "[parameters('masterIgnition')]",
    "linuxConfiguration" : {
     "disablePasswordAuthentication" : true,
     "ssh" : {
      "publicKeys" : [
       {
        "path" : "[variables('sshKeyPath')]",
        "keyData" : "[parameters('sshKeyData')]"
       }
      ]
     }
    }
   },
   "storageProfile" : {
    "imageReference": {
     "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
    },
    "osDisk" : {
```

```
            "name": "[concat(variables('vms').vmNames[copyIndex()].name, '_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "writeAcceleratorEnabled": false,
            "managedDisk": {
              "storageAccountType": "Standard_LRS"
            },
            "diskSizeGB" : "[parameters('diskSizeGB')]"
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vms').vmNames[copyIndex()].name, '-nic'))]",
              "properties": {
                "primary": false
              }
            }
          ]
        },
        "diagnosticsProfile": {
          "bootDiagnostics": {
            "enabled": true,
            "storageUri": "[reference(resourceId('Microsoft.Storage/storageAccounts',
parameters('diagnosticsStorageAccountName'))).primaryEndpoints.blob]"
          }
        }
      }
    }
  ]
}
```

4.2.13. Wait for bootstrap completion and remove bootstrap resources in Azure Stack Hub

After you create all of the required infrastructure in Microsoft Azure Stack Hub, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --name bootstrap_ssh_in
```

```
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
```

```
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
```

```
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
```

```
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-wait --yes
```

```
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-wait
```

```
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name ${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
```

```
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-ssh-pip
```

NOTE

If you do not delete the bootstrap server, installation may not succeed due to API traffic being routed to the bootstrap server.

4.2.14. Creating additional worker machines in Azure Stack Hub

You can create worker machines in Microsoft Azure Stack Hub for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type **06_workers.json** in the file.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster's installation directory. This template describes the worker machines that your cluster requires.

2. Export the following variable needed by the worker machine deployment:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. The Ignition content for the worker nodes.
2. The base name to be used in resource names; this is usually the cluster's infrastructure ID.
3. The name of the storage account for your cluster.

4.2.14.1. ARM template for worker machines

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 4.6. 06_workers.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
```

```
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "metadata" : {
        "description" : "SSH RSA public key file as a string"
      }
    },
    "diagnosticsStorageAccountName": {
      "type": "string"
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_DS4_v2",
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "imageName" : "[parameters('baseName')]",
    "masterAvailabilitySetName" : "[concat(parameters('baseName'), '-cluster')]",
    "numberOfNodes" : "[parameters('numberOfNodes')]",
    "vms" : {
      "copy" : [
        {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfNodes')]",
        "input" : {
            "name" : "[concat(parameters('baseName'), string('-worker-'),
string(copyIndex('vmNames')))]"
        }
      }
    ]
  }
  },
  "resources" : [
    {
      "name": "[parameters('diagnosticsStorageAccountName')]",
      "type": "Microsoft.Storage/storageAccounts",
      "apiVersion": "2017-10-01",
      "location": "[variables('location')]",
      "properties": {},
      "kind": "Storage",
      "sku": {
        "name": "Standard_LRS"
      }
```

```
    },
    {
      "apiVersion" : "2017-10-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "location": "[variables('location')]",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[variables('numberOfNodes')]"
      },
      "name" : "[concat(variables('vms').vmNames[copyIndex()].name, '-nic')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('nodeSubnetRef')]"
              }
            }
          }
        ]
      }
    },
    {
      "apiVersion" : "2017-12-01",
      "type" : "Microsoft.Compute/virtualMachines",
      "location" : "[variables('location')]",
      "copy" : {
        "name" : "vmCopy",
        "count" : "[variables('numberOfNodes')]"
      },
      "name" : "[variables('vms').vmNames[copyIndex()].name]",
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/',
concat(variables('vms').vmNames[copyIndex()].name, '-nic'))]",
        "[concat('Microsoft.Storage/storageAccounts/',
parameters('diagnosticsStorageAccountName'))]"
      ],
      "properties" : {
        "availabilitySet": {
          "id": "
[resourceId('Microsoft.Compute/availabilitySets',variables('masterAvailabilitySetName'))]"
        },
        "hardwareProfile" : {
          "vmSize" : "[parameters('nodeVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vms').vmNames[copyIndex()].name]",
          "adminUsername" : "core",
          "customData" : "[parameters('workerIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : true,
            "ssh" : {
              "publicKeys" : [
                {
```

```
                "path" : "[variables('sshKeyPath')]",
                "keyData" : "[parameters('sshKeyData')]"
              }
            ]
          }
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
        },
        "osDisk" : {
          "name": "[concat(variables('vms').vmNames[copyIndex()].name,'_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "managedDisk": {
            "storageAccountType": "Standard_LRS"
          },
          "diskSizeGB": 128
        }
      },
      "networkProfile" : {
        "networkInterfaces" : [
          {
            "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vms').vmNames[copyIndex()].name, '-nic'))]",
            "properties": {
              "primary": true
            }
          }
        ]
      },
      "diagnosticsProfile": {
        "bootDiagnostics": {
          "enabled": true,
          "storageUri": "[reference(resourceId('Microsoft.Storage/storageAccounts',
parameters('diagnosticsStorageAccountName'))).primaryEndpoints.blob]"
        }
      }
    }
  }
]
}
```

4.2.15. Logging in to the cluster by using the CLI

To log in to your cluster as the default system user, export the **kubeconfig** file. This configuration enables the CLI to authenticate and connect to the specific API server created during OpenShift Container Platform installation.

The **kubeconfig** file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the OpenShift CLI (**oc**).

Procedure

1. Export the **kubeadmin** credentials by running the following command:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

   where:

   <installation_directory>
   Specifies the path to the directory that stores the installation files.

2. Verify you can run **oc** commands successfully using the exported configuration by running the following command:

```
$ oc whoami
```

   Example output

```
system:admin
```

4.2.16. Approving the certificate signing requests for your machines

To add machines to a cluster, verify the status of the certificate signing requests (CSRs) generated for each machine. If manual approval is required, approve the client requests first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

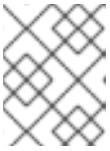
1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

   Example output

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.30.3
master-1  Ready     master  63m  v1.30.3
master-2  Ready     master  64m  v1.30.3
```

   The output lists all of the machines that you created.



NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

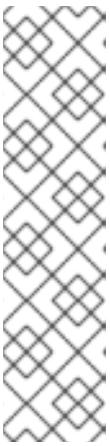
```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.



NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>
```

where:

<csr_name>

Specifies the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>
```

where:

<csr_name>

Specifies the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

Example output

```
NAME      STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.30.3
master-1  Ready    master  73m  v1.30.3
master-2  Ready    master  74m  v1.30.3
worker-0  Ready    worker  11m  v1.30.3
worker-1  Ready    worker  11m  v1.30.3
```

NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

4.2.17. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}.** or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Microsoft Azure Stack Hub by using infrastructure that you provisioned.
- Install the OpenShift CLI (**oc**).
- Install or update the Azure CLI.

Procedure

1. Confirm the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field:

```
$ oc -n openshift-ingress get service router-default
```

Example output

```
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
router-default   LoadBalancer   172.30.20.10    35.130.120.110
80:32288/TCP,443:31215/TCP   20
```

2. Export the Ingress router IP as a variable:

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. Add a ***.apps** record to the DNS zone.

   a. If you are adding this cluster to a new DNS zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

b. If you are adding this cluster to an already existing DNS zone, run:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

Example output

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

4.2.18. Completing an Azure Stack Hub installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure Stack Hub user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure Stack Hub infrastructure.
- Install the **oc** CLI and log in.

Procedure

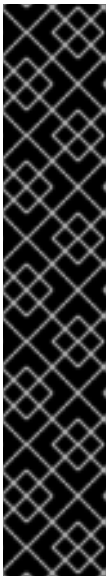
- Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.



IMPORTANT

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

Additional resources

- About remote health monitoring

CHAPTER 5. INSTALLATION CONFIGURATION PARAMETERS FOR AZURE STACK HUB

Before you deploy an OpenShift Container Platform cluster on Azure Stack Hub, you provide a customized **install-config.yaml** installation configuration file that describes the details for your environment.

5.1. AVAILABLE INSTALLATION CONFIGURATION PARAMETERS FOR AZURE STACK HUB

The following tables specify the required, optional, and Azure Stack Hub-specific installation configuration parameters that you can set as part of the installation process.

IMPORTANT

After installation, you cannot change these parameters in the **install-config.yaml** file.

5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.1. Required parameters

Parameter	Description	Values
apiVersion:	The API version for the **install-config.yaml** content. The current version is **v1**. The installation program might also support older API versions.	String
baseDomain:	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the **baseDomain** and **metadata.name** parameter values that uses the **<metadata.name>.<baseDomain>** format.	A fully-qualified domain or subdomain name, such as **example.com**.
metadata:	Kubernetes resource **ObjectMeta**, from which only the **name** parameter is consumed.	Object

Parameter	Description	Values
`metadata:` `  name:`	The name of the cluster. DNS records for the cluster are all subdomains of **{{.metadata.name}}.{{.baseDomain}}**.	String of lowercase letters, hyphens (**-**), and periods (**.**), such as **dev**.
`platform:`	The configuration for the specific platform upon which to perform the installation: **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**. For additional information about **platform.<platform>** parameters, consult the table for your specific platform that follows.	Object
`pullSecret:`	Get a pull secret from Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	`{` `  "auths":{` `  "cloud.openshift.com":{` `  "auth":"b3Blb=",` `  "email":"you@example.com"` `  },` `  "quay.io":{` `  "auth":"b3Blb=",` `  "email":"you@example.com"` `  }` `  }` `}`

5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or configure different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 5.2. Network parameters

Parameter	Description	Values

Parameter	Description	Values
networking:	The configuration for the cluster network.	Object  **NOTE** You cannot change parameters specified by the **networking** object after installation.
networking: networkType:	The Red Hat OpenShift Networking network plugin to install.	**OVNKubernetes**. **OVNKubernetes** is a Container Network Interface (CNI) plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is **OVNKubernetes**.
networking: clusterNetwork:	The IP address blocks for pods. The default value is **10.128.0.0/14** with a host prefix of **/23**. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	Required if you use **networking.clusterNetwork**. An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between **0** and **32**.
networking: clusterNetwork: hostPrefix:	The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23** then each node is assigned a **/23** subnet out of the given**cidr**. A **hostPrefix** value of **23** provides 510 (2^(32 - 23) - 2) pod IP addresses.	A subnet prefix. The default value is **23**.

Parameter	Description	Values
networking: serviceNetwork:	The IP address block for services. The default value is **172.30.0.0/16**. The OVN-Kubernetes network plugins supports only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	Required if you use **networking.machineNetwork**. An IP address block. The default value is **10.0.0.0/16** for all platforms other than libvirt and IBM Power® Virtual Server. For libvirt, the default value is **192.168.126.0/24**. For IBM Power® Virtual Server, the default value is **192.168.0.0/24**.	An IP network block in CIDR notation. For example, **10.0.0.0/16**. **NOTE** Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

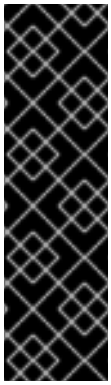
5.1.3. Optional configuration parameters

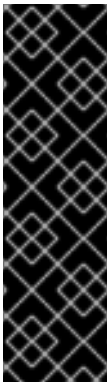
Optional installation configuration parameters are described in the following table:

Table 5.3. Optional parameters

Parameter	Description	Values
additionalTrustBundle:	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle might also be used when a proxy has been configured.	String

Parameter	Description	Values
capabilities:	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in *Installing*.	String array
capabilities: baselineCapabilitySet:	Selects an initial set of optional capabilities to enable. Valid values are **None**, **v4.11**, **v4.12** and **vCurrent**. The default value is **vCurrent**.	String
capabilities: additionalEnabledCapabilities:	Extends the set of optional capabilities beyond what you specify in **baselineCapabilitySet**. You can specify multiple capabilities in this parameter.	String array
cpuPartitioningMode:	Enables workload partitioning, which isolates OpenShift Container Platform services, cluster management workloads, and infrastructure pods to run on a reserved set of CPUs. You can only enable workload partitioning during installation. You cannot disable it after installation. While this field enables workload partitioning, it does not configure workloads to use specific CPUs. For more information, see the *Workload partitioning* page in the *Scalability and Performance* section.	**None** or **AllNodes**. **None** is the default value.
compute:	The configuration for the machines that comprise the compute nodes.	Array of **MachinePool** objects.
compute: architecture:	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are **amd64** (the default).	String

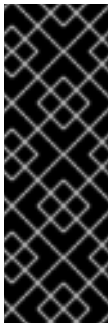
Parameter	Description	Values
compute: hyperthreading:	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  **IMPORTANT** If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
compute: name:	Required if you use **compute**. The name of the machine pool.	**worker**
compute: platform:	Required if you use **compute**. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the **controlPlane.platform** parameter value.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**
compute: replicas:	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to **2**. The default value is **3**.
featureSet:	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as **TechPreviewNoUpgrade**.

Parameter	Description	Values
controlPlane:	The configuration for the machines that form the control plane.	Array of **MachinePool** objects.
controlPlane: architecture:	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are **amd64** (the default).	String
controlPlane: hyperthreading:	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  **IMPORTANT** If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
controlPlane: name:	Required if you use **controlPlane**. The name of the machine pool.	**master**
controlPlane: platform:	Required if you use **controlPlane**. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the **compute.platform** parameter value.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**

Parameter	Description	Values
controlPlane: replicas:	The number of control plane machines to provision.	Supported values are **3**, or **1** when deploying single-node OpenShift.
credentialsMode:	The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. **NOTE** Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the "Managing cloud provider credentials" entry in the *Authentication and authorization* content.	**Mint**, **Passthrough**, **Manual** or an empty string (**""**).
fips:	Enable or disable FIPS mode. The default is **false** (disabled). If you enable FIPS mode, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that RHCOS provides instead.	**false** or **true**

Parameter	Description	Values
	IMPORTANT To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode. When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures. **IMPORTANT** If you are using Azure File storage, you cannot enable FIPS mode.	

Parameter	Description	Values
imageContentSources:	Sources and repositories for the release-image content.	Array of objects. Includes a **source** and, optionally, **mirrors**, as described in the following rows of this table.
imageContentSources: source:	Required if you use **imageContentSources**. Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources: mirrors:	Specify one or more repositories that might also contain the same images.	Array of strings

Parameter	Description	Values
publish:	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	**Internal** or **External**. The default value is **External**. Setting this field to **Internal** is not supported on non-cloud platforms.  **IMPORTANT** If the value of the field is set to **Internal**, the cluster becomes non-functional. For more information, refer to BZ#1953035.
sshKey:	The SSH key to authenticate access to your cluster machines.  **NOTE** For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.	For example, **sshKey: ssh-ed25519 AAAA..**.

5.1.4. Additional Azure Stack Hub configuration parameters

Additional Azure configuration parameters are described in the following table:

Table 5.4. Additional Azure Stack Hub parameters

Parameter	Description	Values
`compute:` `  platform:` `  azure:` `  osDisk:` `  diskSizeGB:`	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **128**.
`compute:` `  platform:` `  azure:` `  osDisk:` `  diskType:`	Defines the type of disk.	**standard_LRS** or **premium_LRS**. The default is **premium_LRS**.
`compute:` `  platform:` `  azure:` `  type:`	Defines the azure instance type for compute machines.	String
`controlPlane:` `  platform:` `  azure:` `  osDisk:` `  diskSizeGB:`	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **1024**.
`controlPlane:` `  platform:` `  azure:` `  osDisk:` `  diskType:`	Defines the type of disk.	**premium_LRS**.
`controlPlane:` `  platform:` `  azure:` `  type:`	Defines the azure instance type for control plane machines.	String
`platform:` `  azure:` `  defaultMachinePlatform:` `  osDisk:` `  diskSizeGB:`	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **128**.

Parameter	Description	Values
`platform:` `  azure:` `  defaultMachinePlatform:` `  osDisk:` `  diskType:`	Defines the type of disk.	**standard_LRS** or **premium_LRS**. The default is **premium_LRS**.
`platform:` `  azure:` `  defaultMachinePlatform:` `  type:`	The Azure instance type for control plane and compute machines.	The Azure instance type.
`platform:` `  azure:` `  armEndpoint:`	The URL of the Azure Resource Manager endpoint that your Azure Stack Hub operator provides.	String
`platform:` `  azure:` `baseDomainResourceGroup Name:`	The name of the resource group that contains the DNS zone for your base domain.	String, for example **production_cluster**.
`platform:` `  azure:` `  region:`	The name of your Azure Stack Hub local region.	String

Parameter	Description	Values
platform: azure: resourceGroupName:	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example **existing_resource_group**.
platform: azure: outboundType:	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available. The outbound routing must be configured before installing a cluster. The installation program does not configure user-defined routing.	**LoadBalancer** or **UserDefinedRouting**. The default is **LoadBalancer**.
platform: azure: cloudName:	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints.	**AzureStackCloud**
clusterOSImage:	The URL of a storage blob in the Azure Stack environment that contains an RHCOS VHD.	String, for example, https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-azurestack.x86_64.vhd

CHAPTER 6. UNINSTALLING A CLUSTER ON AZURE STACK HUB

You can remove a cluster that you deployed to Azure Stack Hub.

6.1. REMOVING A CLUSTER THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE

You can remove a cluster that uses installer-provisioned infrastructure that you provisioned from your cloud platform.

NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

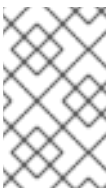
1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info
```

   where:

   <installation_directory>
   Specify the path to the directory that you stored the installation files in.

   --log-level info
   To view different details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.