OpenShift Container Platform 4.19

Installing on Nutanix

Installing OpenShift Container Platform on Nutanix

Last Updated: 2026-03-27

OpenShift Container Platform 4.19 Installing on Nutanix

Installing OpenShift Container Platform on Nutanix

Legal Notice

Abstract

This document describes how to install OpenShift Container Platform on Nutanix.

Table of Contents

CHAPTER 1. INSTALLATION METHODS

You can install an OpenShift Container Platform cluster on Nutanix by using a variety of different installation methods. Each method has qualities that can make the method more suitable for different use cases, such as installing a cluster in a disconnected environment or installing a cluster that requires minimal configuration and provisioning. Before you install OpenShift Container Platform, ensure that your Nutanix environment meets specific requirements.

1.1. NUTANIX VERSION REQUIREMENTS

You must install the OpenShift Container Platform cluster to a Nutanix environment that meets the following requirements:

Table 1.1. Version requirements for Nutanix virtual environments

Component	Required version
Nutanix AOS	6.5.2.7 or later
Prism Central	pc.2022.6 or later

1.2. AGENT-BASED INSTALLER

You can install an OpenShift Container Platform cluster on Nutanix by using the Agent-based Installer. For example, the Agent-based Installer can be used to install a three-node cluster, which is a smaller, more resource efficient cluster for testing, development, and production. See Preparing to install with the Agent-based Installer for additional details.

1.3. ENVIRONMENT REQUIREMENTS

Before you install an OpenShift Container Platform cluster, review the following Nutanix AOS environment requirements.

1.3.1. Infrastructure requirements

You can install OpenShift Container Platform on on-premise Nutanix clusters, Nutanix Cloud Clusters (NC2) on Amazon Web Services (AWS), or NC2 on Microsoft Azure.

For more information, see Nutanix Cloud Clusters on AWS and Nutanix Cloud Clusters on Microsoft Azure.

1.3.2. Required account privileges

The installation program requires access to a Nutanix account with the necessary permissions to deploy the cluster and to maintain the daily operation of it. The following options are available to you:

- You can use a local Prism Central user account with administrative privileges. Using a local account is the quickest way to grant access to an account with the required permissions.
- If your organization's security policies require that you use a more restrictive set of permissions, use the permissions that are listed in the following table to create a custom Cloud Native role in Prism Central. You can then assign the role to a user account that is a member of a Prism

Central authentication directory.

Consider the following when managing this user account:

- When assigning entities to the role, ensure that the user can access only the Prism Element and subnet that are required to deploy the virtual machines.
- Ensure that the user is a member of the project to which it needs to assign virtual machines.

For more information, see the Nutanix documentation about creating a Custom Cloud Native role, assigning a role, and adding a user to a project.

Example 1.1. Required permissions for creating a Custom Cloud Native role

Nutanix Object	When required	Required permissions in Nutanix API	Description
Categories	Always	**Create_Category_Mapping** **Create_Or_Update_Name_Category** **Create_Or_Update_Value_Category** **Delete_Category_Mapping** **Delete_Name_Category** **Delete_Value_Category** **View_Category_Mapping** **View_Name_Category** **View_Value_Category**	Create, read, and delete categories that are assigned to the OpenShift Container Platform machines.
Images	Always	**Create_Image** **Delete_Image** **View_Image**	Create, read, and delete the operating system images used for the OpenShift Container Platform machines.
Virtual Machines	Always	**Create_Virtual_Machine** **Delete_Virtual_Machine** **View_Virtual_Machine**	Create, read, and delete the OpenShift Container Platform machines.

Nutanix Object	When required	Required permissions in Nutanix API	Description
Clusters	Always	**View_Cluster**	View the Prism Element clusters that host the OpenShift Container Platform machines.
Subnets	Always	**View_Subnet**	View the subnets that host the OpenShift Container Platform machines.
Projects	If you will associate a project with compute machines, control plane machines, or all machines.	**View_Project**	View the projects defined in Prism Central and allow a project to be assigned to the OpenShift Container Platform machines.
Tasks	Always	**View_Task**	Fetch and view tasks on the Prism Element that contain OpenShift Container Platform machines and nodes.
Hosts	If you use GPUs with compute machines.	**View_Host**	Fetch and view hosts on the Prism Element that have GPUs attached.

1.3.3. Cluster limits

Available resources vary between clusters. The number of possible clusters within a Nutanix environment is limited primarily by available storage space and any limitations associated with the resources that the cluster creates, and resources that you require to deploy the cluster, such a IP addresses and networks.

1.3.4. Cluster resources

A minimum of 800 GB of storage is required to use a standard cluster.

When you deploy a OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your Nutanix instance. Although these resources use 856 GB of storage, the bootstrap node is destroyed as part of the installation process.

A standard OpenShift Container Platform installation creates the following resources:

- 1 label

- Virtual machines:
  - 1 disk image
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

1.3.5. Networking requirements

You must use either AHV IP Address Management (IPAM) or Dynamic Host Configuration Protocol (DHCP) for the network and ensure that it is configured to provide persistent IP addresses to the cluster machines. Additionally, create the following networking resources before you install the OpenShift Container Platform cluster:

- IP addresses
- DNS records

Nutanix Flow Virtual Networking is supported for new cluster installations. To use this feature, enable Flow Virtual Networking on your AHV cluster before installing. For more information, see Flow Virtual Networking overview.

NOTE

It is recommended that each OpenShift Container Platform node in the cluster have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, an NTP server prevents errors typically associated with asynchronous server clocks.

1.3.5.1. Required IP Addresses

An installer-provisioned installation requires two static virtual IP (VIP) addresses:

- A VIP address for the API is required. This address is used to access the cluster API.
- A VIP address for ingress is required. This address is used for cluster ingress traffic.

You specify these IP addresses when you install the OpenShift Container Platform cluster.

1.3.5.2. DNS records

You must create DNS records for two static IP addresses in the appropriate DNS server for the Nutanix instance that hosts your OpenShift Container Platform cluster. In each record, **<cluster_name>** is the cluster name and **<base_domain>** is the cluster base domain that you specify when you install the cluster.

If you use your own DNS or DHCP server, you must also create records for each node, including the bootstrap, control plane, and compute nodes.

A complete DNS record takes the form: **<component>.<cluster_name>.<base_domain>.**.

Table 1.2. Required DNS records

Component	Record	Description
API VIP	**api.<cluster_name>.<base_domain>.**	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

1.4. CONFIGURING THE CLOUD CREDENTIAL OPERATOR UTILITY

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). To install a cluster on Nutanix, you must set the CCO to **manual** mode as part of the installation process.

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

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

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

NOTE

Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

NOTE

The **ccoctl** binary is created in the directory from where you executed the command and not in **/usr/bin/**. You must rename the directory or move the **ccoctl.<rhel_version>** binary to **ccoctl**.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
```

```
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

Additional resources

- Preparing to update a cluster with manually maintained credentials

CHAPTER 2. FAULT TOLERANT DEPLOYMENTS USING MULTIPLE PRISM ELEMENTS

By default, the installation program installs control plane and compute machines into a single Nutanix Prism Element (cluster). To improve the fault tolerance of your OpenShift Container Platform cluster, you can specify that these machines be distributed across multiple Nutanix clusters by configuring failure domains.

A failure domain represents an additional Prism Element instance that is available to OpenShift Container Platform machine pools during and after installation.

2.1. INSTALLATION METHOD AND FAILURE DOMAIN CONFIGURATION

The OpenShift Container Platform installation method determines how and when you configure failure domains:

- If you deploy using installer-provisioned infrastructure, you can configure failure domains in the installation configuration file before deploying the cluster. For more information, see Configuring failure domains.
  You can also configure failure domains after the cluster is deployed. For more information about configuring failure domains post-installation, see Adding failure domains to an existing Nutanix cluster.

- If you deploy using infrastructure that you manage (user-provisioned infrastructure) no additional configuration is required. After the cluster is deployed, you can manually distribute control plane and compute machines across failure domains.

2.2. ADDING FAILURE DOMAINS TO AN EXISTING NUTANIX CLUSTER

By default, the installation program installs control plane and compute machines into a single Nutanix Prism Element (cluster). After an OpenShift Container Platform cluster is deployed, you can improve its fault tolerance by adding additional Prism Element instances to the deployment using failure domains.

A failure domain represents a single Prism Element instance where new control plane and compute machines can be deployed and existing control plane and compute machines can be distributed.

2.2.1. Failure domain requirements

When planning to use failure domains, consider the following requirements:

- All Nutanix Prism Element instances must be managed by the same instance of Prism Central. A deployment that is comprised of multiple Prism Central instances is not supported.

- The machines that make up the Prism Element clusters must reside on the same Ethernet network for failure domains to be able to communicate with each other.

- A subnet is required in each Prism Element that will be used as a failure domain in the OpenShift Container Platform cluster. When defining these subnets, they must share the same IP address prefix (CIDR) and should contain the virtual IP addresses that the OpenShift Container Platform cluster uses.

2.2.2. Adding failure domains to the Infrastructure CR

You add failure domains to an existing Nutanix cluster by modifying its Infrastructure custom resource (CR) (**infrastructures.config.openshift.io**).

TIP

To ensure high-availability, configure three failure domains.

Procedure

1. Edit the Infrastructure CR by running the following command:

```
$ oc edit infrastructures.config.openshift.io cluster
```

2. Configure the failure domains.

Example Infrastructure CR with Nutanix failure domains

```
spec:
  cloudConfig:
    key: config
    name: cloud-provider-config
#...
  platformSpec:
    nutanix:
      failureDomains:
      - cluster:
          type: UUID
          uuid: <uuid>
        name: <failure_domain_name>
        subnets:
        - type: UUID
          uuid: <network_uuid>
      - cluster:
          type: UUID
          uuid: <uuid>
        name: <failure_domain_name>
        subnets:
        - type: UUID
          uuid: <network_uuid>
      - cluster:
          type: UUID
          uuid: <uuid>
        name: <failure_domain_name>
        subnets:
        - type: UUID
          uuid: <network_uuid>
# ...
```

where:

<uuid>

Specifies the universally unique identifier (UUID) of the Prism Element.

<failure_domain_name>

Specifies a unique name for the failure domain. The name is limited to 64 or fewer characters, which can include lower-case letters, digits, and a dash (**-**). The dash cannot be in the leading or ending position of the name.

<network_uuid>

Specifies one or more UUID for the Prism Element subnet object. The CIDR IP address prefix for one of the specified subnets must contain the virtual IP addresses that the OpenShift Container Platform cluster uses.

IMPORTANT

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

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

To configure multiple subnets in the Infrastructure CR, you must enable the **NutanixMultiSubnets** feature gate. A maximum of 32 subnets for each failure domain (Prism Element) in an OpenShift Container Platform cluster is supported. All subnet UUID values must be unique.

3. Save the CR to apply the changes.

2.2.3. Distributing control planes across failure domains

You distribute control planes across Nutanix failure domains by modifying the control plane machine set custom resource (CR).

Prerequisites

- You have configured the failure domains in the cluster's Infrastructure custom resource (CR).
- The control plane machine set custom resource (CR) is in an active state.

For more information on checking the control plane machine set custom resource state, see "Additional resources".

Procedure

1. Edit the control plane machine set CR by running the following command:

```
$ oc edit controlplanemachineset.machine.openshift.io cluster -n openshift-machine-api
```

2. Configure the control plane machine set to use failure domains by adding a **spec.template.machines_v1beta1_machine_openshift_io.failureDomains** stanza.

Example control plane machine set with Nutanix failure domains

```
apiVersion: machine.openshift.io/v1
kind: ControlPlaneMachineSet
  metadata:
    creationTimestamp: null
    labels:
      machine.openshift.io/cluster-api-cluster: <cluster_name>
    name: cluster
    namespace: openshift-machine-api
spec:
# ...
  template:
    machineType: machines_v1beta1_machine_openshift_io
    machines_v1beta1_machine_openshift_io:
      failureDomains:
        platform: Nutanix
        nutanix:
        - name: <failure_domain_name_1>
        - name: <failure_domain_name_2>
        - name: <failure_domain_name_3>
# ...
```

3. Save your changes.

By default, the control plane machine set propagates changes to your control plane configuration automatically. If the cluster is configured to use the **OnDelete** update strategy, you must replace your control planes manually. For more information, see "Additional resources".

Additional resources

- Checking the control plane machine set custom resource state
- Replacing a control plane machine

2.2.4. Distributing compute machines across failure domains

You can distribute compute machines across Nutanix failure domains one of the following ways:

- Editing existing compute machine sets allows you to distribute compute machines across Nutanix failure domains as a minimal configuration update.
- Replacing existing compute machine sets ensures that the specification is immutable and all your machines are the same.

2.2.4.1. Editing compute machine sets to implement failure domains

To distribute compute machines across Nutanix failure domains by using an existing compute machine set, you update the compute machine set with your configuration and then use scaling to replace the existing compute machines.

Prerequisites

- You have configured the failure domains in the cluster's Infrastructure custom resource (CR).

Procedure

1. Run the following command to view the cluster's Infrastructure CR.

```
$ oc describe infrastructures.config.openshift.io cluster
```

2. For each failure domain (**platformSpec.nutanix.failureDomains**), note the cluster's UUID, name, and subnet object UUID. These values are required to add a failure domain to a compute machine set.

3. List the compute machine sets in your cluster by running the following command:

```
$ oc get machinesets -n openshift-machine-api
```

Example output

```
NAME                  DESIRED   CURRENT   READY   AVAILABLE   AGE
<machine_set_name_1>  1         1         1       1           55m
<machine_set_name_2>  1         1         1       1           55m
```

4. Edit the first compute machine set by running the following command:

```
$ oc edit machineset <machine_set_name_1> -n openshift-machine-api
```

5. Configure the compute machine set to use the first failure domain by updating the following to the **spec.template.spec.providerSpec.value** stanza.

NOTE

Be sure that the values you specify for the **cluster** and **subnets** fields match the values that were configured in the **failureDomains** stanza in the cluster's Infrastructure CR.

Example compute machine set with Nutanix failure domains

```
apiVersion: machine.openshift.io/v1
kind: MachineSet
metadata:
  creationTimestamp: null
  labels:
    machine.openshift.io/cluster-api-cluster: <cluster_name>
  name: <machine_set_name_1>
  namespace: openshift-machine-api
spec:
  replicas: 2
# ...
  template:
    spec:
# ...
      providerSpec:
        value:
          apiVersion: machine.openshift.io/v1
          failureDomain:
            name: <failure_domain_name_1>
          cluster:
```

```
          type: uuid
          uuid: <prism_element_uuid_1>
        subnets:
        - type: uuid
          uuid: <prism_element_network_uuid_1>
# ...
```

6. Note the value of **spec.replicas**, because you need it when scaling the compute machine set to apply the changes.

7. Save your changes.

8. List the machines that are managed by the updated compute machine set by running the following command:

```
$ oc get -n openshift-machine-api machines \
  -l machine.openshift.io/cluster-api-machineset=<machine_set_name_1>
```

Example output

```
NAME                      PHASE     TYPE   REGION    ZONE            AGE
<machine_name_original_1>  Running   AHV    Unnamed   Development-STS   4h
<machine_name_original_2>  Running   AHV    Unnamed   Development-STS   4h
```

9. For each machine that is managed by the updated compute machine set, set the **delete** annotation by running the following command:

```
$ oc annotate machine/<machine_name_original_1> \
  -n openshift-machine-api \
  machine.openshift.io/delete-machine="true"
```

10. To create replacement machines with the new configuration, scale the compute machine set to twice the number of replicas by running the following command:

```
$ oc scale --replicas=<twice_the_number_of_replicas> \ (1)
  machineset <machine_set_name_1> \
  -n openshift-machine-api
```

(1) For example, if the original number of replicas in the compute machine set is **2**, scale the replicas to **4**.

11. List the machines that are managed by the updated compute machine set by running the following command:

```
$ oc get -n openshift-machine-api machines -l machine.openshift.io/cluster-api-machineset=
<machine_set_name_1>
```

When the new machines are in the **Running** phase, you can scale the compute machine set to the original number of replicas.

12. To remove the machines that were created with the old configuration, scale the compute machine set to the original number of replicas by running the following command:

```
$ oc scale --replicas=<original_number_of_replicas> \ (1)
  machineset <machine_set_name_1> \
  -n openshift-machine-api
```

(1) For example, if the original number of replicas in the compute machine set was **2**, scale the replicas to **2**.

13. As required, continue to modify machine sets to reference the additional failure domains that are available to the deployment.

Additional resources

- Modifying a compute machine set

2.2.4.2. Replacing compute machine sets to implement failure domains

To distribute compute machines across Nutanix failure domains by replacing a compute machine set, you create a new compute machine set with your configuration, wait for the machines that it creates to start, and then delete the old compute machine set.

Prerequisites

- You have configured the failure domains in the cluster's Infrastructure custom resource (CR).

Procedure

1. Run the following command to view the cluster's Infrastructure CR.

```
$ oc describe infrastructures.config.openshift.io cluster
```

2. For each failure domain (**platformSpec.nutanix.failureDomains**), note the cluster's UUID, name, and subnet object UUID. These values are required to add a failure domain to a compute machine set.

3. List the compute machine sets in your cluster by running the following command:

```
$ oc get machinesets -n openshift-machine-api
```

Example output

```
NAME                        DESIRED  CURRENT  READY  AVAILABLE  AGE
<original_machine_set_name_1>  1        1        1      1          55m
<original_machine_set_name_2>  1        1        1      1          55m
```

4. Note the names of the existing compute machine sets.

5. Create a YAML file that contains the values for your new compute machine set custom resource (CR) by using one of the following methods:

   - Copy an existing compute machine set configuration into a new file by running the following command:

```
$ oc get machineset <original_machine_set_name_1> \
  -n openshift-machine-api -o yaml > <new_machine_set_name_1>.yaml
```

You can edit this YAML file with your preferred text editor.

- Create a blank YAML file named **<new_machine_set_name_1>.yaml** with your preferred text editor and include the required values for your new compute machine set.
If you are not sure which value to set for a specific field, you can view values of an existing compute machine set CR by running the following command:

```
$ oc get machineset <original_machine_set_name_1> \
  -n openshift-machine-api -o yaml
```

Example output

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <infrastructure_id>-<role> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id>
      machine.openshift.io/cluster-api-machineset: <infrastructure_id>-<role>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id>
        machine.openshift.io/cluster-api-machine-role: <role>
        machine.openshift.io/cluster-api-machine-type: <role>
        machine.openshift.io/cluster-api-machineset: <infrastructure_id>-<role>
    spec:
      providerSpec: 3
        ...
```

1. The cluster infrastructure ID.
2. A default node label.

NOTE

For clusters that have user-provisioned infrastructure, a compute machine set can only create machines with a **worker** or **infra** role.

3. The values in the **<providerSpec>** section of the compute machine set CR are platform-specific. For more information about **<providerSpec>** parameters in the CR, see the sample compute machine set CR configuration for your provider.

6. Configure the new compute machine set to use the first failure domain by updating or adding the following to the **spec.template.spec.providerSpec.value** stanza in the **<new_machine_set_name_1>.yaml** file.

NOTE

Be sure that the values you specify for the **cluster** and **subnets** fields match the values that were configured in the **failureDomains** stanza in the cluster's Infrastructure CR.

Example compute machine set with Nutanix failure domains

```
apiVersion: machine.openshift.io/v1
kind: MachineSet
metadata:
  creationTimestamp: null
  labels:
    machine.openshift.io/cluster-api-cluster: <cluster_name>
  name: <new_machine_set_name_1>
  namespace: openshift-machine-api
spec:
  replicas: 2
# ...
  template:
    spec:
# ...
      providerSpec:
        value:
          apiVersion: machine.openshift.io/v1
          failureDomain:
            name: <failure_domain_name_1>
          cluster:
            type: uuid
            uuid: <prism_element_uuid_1>
          subnets:
          - type: uuid
            uuid: <prism_element_network_uuid_1>
# ...
```

7. Save your changes.

8. Create a compute machine set CR by running the following command:

```
$ oc create -f <new_machine_set_name_1>.yaml
```

9. As required, continue to create compute machine sets to reference the additional failure domains that are available to the deployment.

10. List the machines that are managed by the new compute machine sets by running the following command for each new compute machine set:

```
$ oc get -n openshift-machine-api machines -l machine.openshift.io/cluster-api-machineset=
<new_machine_set_name_1>
```

Example output

```
NAME                      PHASE         TYPE   REGION    ZONE            AGE
<machine_from_new_1>          Provisioned   AHV    Unnamed   Development-STS   25s
<machine_from_new_2>          Provisioning  AHV    Unnamed   Development-STS   25s
```

When the new machines are in the **Running** phase, you can delete the old compute machine sets that do not include the failure domain configuration.

11. When you have verified that the new machines are in the **Running** phase, delete the old compute machine sets by running the following command for each:

```
$ oc delete machineset <original_machine_set_name_1> -n openshift-machine-api
```

Verification

- To verify that the compute machine sets without the updated configuration are deleted, list the compute machine sets in your cluster by running the following command:

```
$ oc get machinesets -n openshift-machine-api
```

Example output

```
NAME                  DESIRED   CURRENT   READY   AVAILABLE   AGE
<new_machine_set_name_1>  1         1         1       1           4m12s
<new_machine_set_name_2>  1         1         1       1           4m12s
```

- To verify that the compute machines without the updated configuration are deleted, list the machines in your cluster by running the following command:

```
$ oc get -n openshift-machine-api machines
```

Example output while deletion is in progress

```
NAME                    PHASE         TYPE     REGION     ZONE            AGE
<machine_from_new_1>        Running       AHV      Unnamed    Development-STS   5m41s
<machine_from_new_2>        Running       AHV      Unnamed    Development-STS   5m41s
<machine_from_original_1>   Deleting      AHV      Unnamed    Development-STS   4h
<machine_from_original_2>   Deleting      AHV      Unnamed    Development-STS   4h
```

Example output when deletion is complete

```
NAME                    PHASE         TYPE     REGION     ZONE            AGE
<machine_from_new_1>        Running       AHV      Unnamed    Development-STS   6m30s
<machine_from_new_2>        Running       AHV      Unnamed    Development-STS   6m30s
```

- To verify that a machine created by the new compute machine set has the correct configuration, examine the relevant fields in the CR for one of the new machines by running the following command:

```
$ oc describe machine <machine_from_new_1> -n openshift-machine-api
```

Additional resources

- Creating a compute machine set on Nutanix

2.3. IMPROVING RELIABILITY FOR MULTIPLE SUBNET CONFIGURATIONS ON NUTANIX

To improve reliability and avoid common networking problems with multiple subnet configurations on Nutanix, adhere to the configuration practices that minimize networking conflicts.

The following networking configuration and management practices can help your multiple subnet configuration perform more reliably:

- To avoid overlapping IP address assignments, use predefined static IP addresses in the **cloud-init** metadata.
- Tag all VMs, disks, and networks with a unique cluster ID.
- Avoid IP address conflicts by using dedicated subnets for each OpenShift Container Platform cluster:
  Nutanix uses Nutanix Acropolis Hypervisor (AHV) and Nutanix Prism networking to assign IP addresses to virtual machines (VMs). If a single subnet provides IP addresses for more than one OpenShift Container Platform cluster, AHV or Prism might assign the same IP address to a VM or pod in more than one cluster.

  To avoid this issue, use dedicated subnets for each OpenShift Container Platform cluster, even when you have more than one cluster on a single Prism Central instance. You can use the Prism UI or automation tools, such as Terraform or Ansible, to create separate IP address pools for each OpenShift Container Platform cluster.
- Ensure that each OpenShift Container Platform cluster uses distinct DNS zones and virtual IP address ranges.
- Avoid DHCP conflicts by maintaining DHCP allocations:
  If you use Nutanix to manage DHCP allocation, objects in your cluster might have duplicate leases. Duplicate leases can cause DHCP conflicts when you apply changes to the control plane machine set custom resource (CR) specification.

  To avoid this issue, regularly remove stale DHCP leases.
- Use automation tools, such as Terraform or Ansible, to isolate the infrastructure for each OpenShift Container Platform cluster.

CHAPTER 3. INSTALLING A CLUSTER ON NUTANIX

In OpenShift Container Platform version 4.19, you can choose one of the following options to install a cluster on your Nutanix instance:

Using installer-provisioned infrastructure: Use the procedures in the following sections to use installer-provisioned infrastructure. Installer-provisioned infrastructure is ideal for installing in connected or disconnected network environments. The installer-provisioned infrastructure includes an installation program that provisions the underlying infrastructure for the cluster.

Using the Assisted Installer: The Assisted Installer hosted at console.redhat.com. The Assisted Installer cannot be used in disconnected environments. The Assisted Installer does not provision the underlying infrastructure for the cluster, so you must provision the infrastructure before you run the Assisted Installer. Installing with the Assisted Installer also provides integration with Nutanix, enabling autoscaling. See Installing an on-premise cluster using the Assisted Installer for additional details.

Using user-provisioned infrastructure: Complete the relevant steps outlined in the Installing a cluster on any platform documentation.

3.1. PREREQUISITES

- You have reviewed details about the OpenShift Container Platform installation and update processes.
- The installation program requires access to port 9440 on Prism Central and Prism Element. You verified that port 9440 is accessible.
- If you use a firewall, you have met these prerequisites:
  - You confirmed that port 9440 is accessible. Control plane nodes must be able to reach Prism Central and Prism Element on port 9440 for the installation to succeed.
  - You configured the firewall to grant access to the sites that OpenShift Container Platform requires. This includes the use of Telemetry.
- If your Nutanix environment is using the default self-signed SSL certificate, replace it with a certificate that is signed by a CA. The installation program requires a valid CA-signed certificate to access to the Prism Central API. For more information about replacing the self-signed certificate, see the Nutanix AOS Security Guide .
  If your Nutanix environment uses an internal CA to issue certificates, you must configure a cluster-wide proxy as part of the installation process. For more information, see Configuring a custom PKI.

IMPORTANT

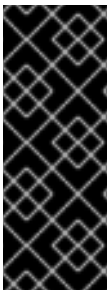
Use 2048-bit certificates. The installation fails if you use 4096-bit certificates with Prism Central 2022.x.

3.2. INTERNET ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.19, 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.

3.3. INTERNET ACCESS FOR PRISM CENTRAL

Prism Central requires internet access to obtain the Red Hat Enterprise Linux CoreOS (RHCOS) image that is required to install the cluster. The RHCOS image for Nutanix is available at **rhcos.mirror.openshift.com**.

3.4. 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.

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.

3.5. 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.

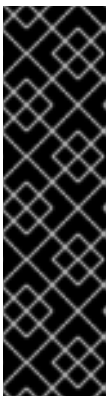
Procedure

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.

3.6. ADDING NUTANIX ROOT CA CERTIFICATES TO YOUR SYSTEM TRUST

Because the installation program requires access to the Prism Central API, you must add your Nutanix trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the Prism Central web console, download the Nutanix root CA certificates.
2. Extract the compressed file that contains the Nutanix root CA certificates.
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

   ```
   # cp certs/lin/* /etc/pki/ca-trust/source/anchors
   ```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

   ```
   # update-ca-trust extract
   ```

3.7. CREATING THE INSTALLATION CONFIGURATION FILE

You can customize the OpenShift Container Platform cluster you install on

Nutanix.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have verified that you have met the Nutanix networking requirements. For more information, see "Preparing to install on Nutanix".

Procedure

1. Create the **install-config.yaml** file.

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

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

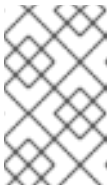
- **<installation_directory>**: For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
  When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.

- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore 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.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access 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.

ii. Select **nutanix** as the platform to target.

iii. Enter the Prism Central domain name or IP address.

iv. Enter the port that is used to log into Prism Central.

v. Enter the credentials that are used to log into Prism Central.
The installation program connects to Prism Central.

vi. Select the Prism Element that will manage the OpenShift Container Platform cluster.

vii. Select the network subnet to use.

viii. Enter the virtual IP address that you configured for control plane API access.

ix. Enter the virtual IP address that you configured for cluster ingress.

x. Enter the base domain. This base domain must be the same one that you configured in the DNS records.

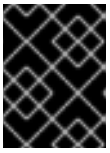
xi. Enter a descriptive name for your cluster.
The cluster name you enter must match the cluster name you specified when configuring the DNS records.

2. Optional: Update one or more of the default configuration parameters in the **install.config.yaml** file to customize the installation.
For more information about the parameters, see "Installation configuration parameters".

NOTE

If you are installing a three-node cluster, be sure to set the **compute.replicas** parameter to **0**. This ensures that cluster's control planes are schedulable. For more information, see "Installing a three-node cluster on Nutanix".

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



IMPORTANT

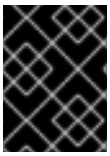
The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- Installation configuration parameters for Nutanix

3.7.1. Sample customized install-config.yaml file for Nutanix

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. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    nutanix: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
      categories: (5)
      - key: <category_key_name>
        value: <category_value>
controlPlane: (6)
  hyperthreading: Enabled (7)
  name: master
  replicas: 3
  platform:
    nutanix: (8)
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
```

```
      osDisk:
        diskSizeGiB: 120
      categories: 9
      - key: <category_key_name>
        value: <category_value>
metadata:
  creationTimestamp: null
  name: test-cluster 10
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
    - 172.30.0.0/16
platform:
  nutanix:
    apiVIPs:
      - 10.40.142.7 12
    defaultMachinePlatform:
      bootType: Legacy
      categories: 13
      - key: <category_key_name>
        value: <category_value>
      project: 14
        type: name
        name: <project_name>
    ingressVIPs:
      - 10.40.142.8 15
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 16
        port: 9440 17
      password: <password> 18
      username: <username> 19
    prismElements:
    - endpoint:
        address: your.prismelement.domainname
        port: 9440
      uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
    subnetUUIDs:
    - c7938dc6-7659-453e-a688-e26020c68e43
    clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
20
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
```

1 10 12 15 16 17 18 19 21 Required. The installation program prompts you for this value.

2 6 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.

3 7 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4 8 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

5 9 13 Optional: Provide one or more pairs of a prism category key and a prism category value. These category key-value pairs must exist in Prism Central. You can provide separate categories to compute machines, control plane machines, or all machines.

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

14 Optional: Specify a project with which VMs are associated. Specify either **name** or **uuid** for the project type, and then provide the corresponding UUID or project name. You can associate projects to compute machines, control plane machines, or all machines.

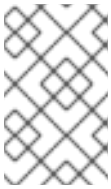
20 Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server and pointing the installation program to the image.

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 Optional: You can 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.

3.7.2. Configuring failure domains

Failure domains improve the fault tolerance of an OpenShift Container Platform cluster by distributing control plane and compute machines across multiple Nutanix Prism Elements (clusters).

TIP

It is recommended that you configure three failure domains to ensure high-availability.

Prerequisites

- You have an installation configuration file (**install-config.yaml**).

Procedure

1. Edit the **install-config.yaml** file and add the following stanza to configure the first failure domain:

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
platform:
  nutanix:
    failureDomains:
    - name: <failure_domain_name>
      prismElement:
        name: <prism_element_name>
        uuid: <prism_element_uuid>
      subnetUUIDs:
      - <network_uuid>
# ...
```

where:

<failure_domain_name>

Specifies a unique name for the failure domain. The name is limited to 64 or fewer characters, which can include lower-case letters, digits, and a dash (**-**). The dash cannot be in the leading or ending position of the name.

<prism_element_name>

Optional. Specifies the name of the Prism Element.

<prism_element_uuid>

Specifies the UUID of the Prism Element.

<network_uuid>

Specifies the one or more UUIDs of the Prism Element subnet objects. Among them, one of the subnet's IP address prefixes (CIDRs) must contain the virtual IP addresses that the OpenShift Container Platform cluster uses. A maximum of 32 subnets for each failure domain (Prism Element) in an OpenShift Container Platform cluster is supported. All **subnetUUID** values must be unique.

2. As required, configure additional failure domains.

3. To distribute control plane and compute machines across the failure domains, do one of the following:

   - If compute and control plane machines can share the same set of failure domains, add the failure domain names under the cluster's default machine configuration.

     Example of control plane and compute machines sharing a set of failure domains

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     platform:
       nutanix:
         defaultMachinePlatform:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     ```

   - If compute and control plane machines must use different failure domains, add the failure domain names under the respective machine pools.

     Example of control plane and compute machines using different failure domains

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     controlPlane:
       platform:
         nutanix:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     compute:
       platform:
         nutanix:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
     # ...
     ```

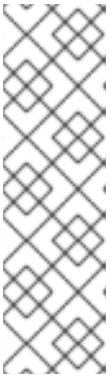
4. Save the file.

3.7.3. 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.

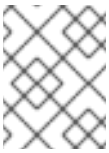
For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

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.

3.8. 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.

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 v4.19 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>
```

3.9. 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 v4.19 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>
```

3.10. 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 v4.19 macOS Clients** entry and save the file.

NOTE

For macOS arm64, choose the **OpenShift v4.19 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>
```

3.11. CONFIGURING IAM FOR NUTANIX

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an **install-config.yaml** file.

Procedure

1. Create a YAML file that contains the credentials data in the following format:

   Credentials data format

   ```
   credentials:
   - type: basic_auth (1)
     data:
       prismCentral: (2)
         username: <username_for_prism_central>
         password: <password_for_prism_central>
       prismElements: (3)
       - name: <name_of_prism_element>
         username: <username_for_prism_element>
         password: <password_for_prism_element>
   ```

   (1) Specify the authentication type. Only basic authentication is supported.

   (2) Specify the Prism Central credentials.

   (3) Optional: Specify the Prism Element credentials.

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.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
  secretRef:
    name: nutanix-credentials
    namespace: openshift-machine-api
```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

```
$ ccoctl nutanix create-shared-secrets \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \1
  --output-dir=<ccoctl_output_dir> \2
  --credentials-source-filepath=<path_to_credentials_file> 3
```

1 Specify the path to the directory that contains the files for the component **CredentialsRequests** objects.

2 Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.

3 Optional: Specify the directory that contains the credentials data YAML file. By default, **ccoctl** expects this file to be in **<home_directory>/.nutanix/credentials**.

5. Edit the **install-config.yaml** configuration file so that the **credentialsMode** parameter is set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
...
```

1 Add this line to set the **credentialsMode** parameter to **Manual**.

6. Create the installation manifests by running the following command:

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

1 Specify the path to the directory that contains the **install-config.yaml** file for your cluster.

7. Copy the generated credential files to the target manifests directory by running the following command:

```
$ cp <ccoctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

Verification

- Ensure that the appropriate secrets exist in the **manifests** directory.

```
$ ls ./<installation_directory>/manifests
```

Example output

```
cluster-config.yaml
cluster-dns-02-config.yml
cluster-infrastructure-02-config.yml
cluster-ingress-02-config.yml
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-proxy-01-config.yaml
cluster-scheduler-02-config.yml
cvo-overrides.yaml
kube-cloud-config.yaml
kube-system-configmap-root-ca.yaml
machine-config-server-tls-secret.yaml
openshift-config-secret-pull-secret.yaml
openshift-cloud-controller-manager-nutanix-credentials-credentials.yaml
openshift-machine-api-nutanix-credentials-credentials.yaml
```

3.12. ADDING CONFIG MAP AND SECRET RESOURCES REQUIRED FOR NUTANIX CCM

Installations on Nutanix require additional **ConfigMap** and **Secret** resources to integrate with the Nutanix Cloud Controller Manager (CCM).

Prerequisites

- You have created a **manifests** directory within your installation directory.

Procedure

1. Navigate to the **manifests** directory:

-

```
$ cd <path_to_installation_directory>/manifests
```

2. Create the **cloud-conf ConfigMap** file with the name **openshift-cloud-controller-manager-cloud-config.yaml** and add the following information:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
data:
  cloud.conf: "{
     \"prismCentral\": {
        \"address\": \"<prism_central_FQDN/IP>\", (1)
        \"port\": 9440,
         \"credentialRef\": {
            \"kind\": \"Secret\",
            \"name\": \"nutanix-credentials\",
            \"namespace\": \"openshift-cloud-controller-manager\"
         }
      },
      \"topologyDiscovery\": {
          \"type\": \"Prism\",
          \"topologyCategories\": null
      },
      \"enableCustomLabeling\": true
    }"
```

(1) Specify the Prism Central FQDN/IP.

3. Verify that the file **cluster-infrastructure-02-config.yml** exists and has the following information:

```
spec:
  cloudConfig:
    key: config
    name: cloud-provider-config
```

3.13. SERVICES FOR A USER-MANAGED LOAD BALANCER

To integrate your infrastructure with existing network standards or gain more control over traffic management in OpenShift Container Platform , configure services for a user-managed load balancer.

IMPORTANT

Configuring a user-managed load balancer depends on your vendor's load balancer.

The information and examples in this section are for guideline purposes only. Consult the vendor documentation for more specific information about the vendor's load balancer.

Red Hat supports the following services for a user-managed load balancer:

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

You can choose whether you want to configure one or all of these services for a user-managed load balancer. Configuring only the Ingress Controller service is a common configuration option. To better understand each service, view the following diagrams:

Figure 3.1. Example network workflow that shows an Ingress Controller operating in an OpenShift Container Platform environment

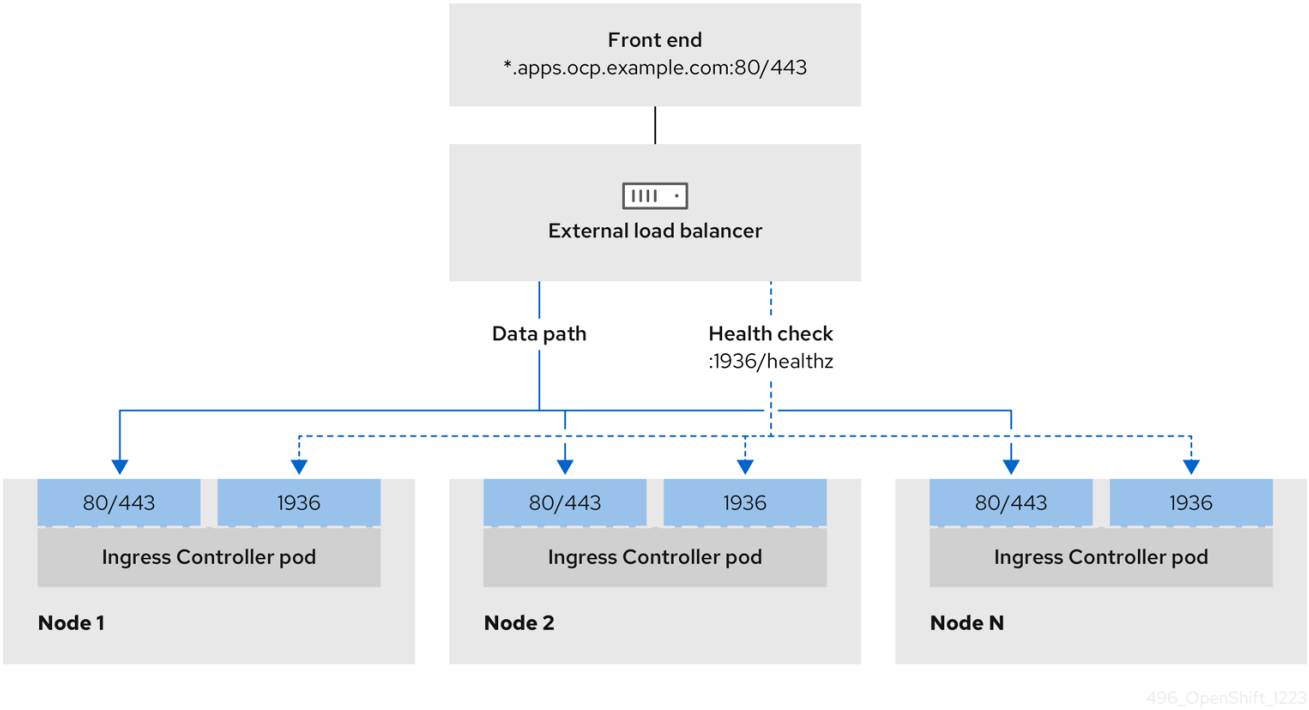


Figure 3.2. Example network workflow that shows an OpenShift API operating in an OpenShift Container Platform environment

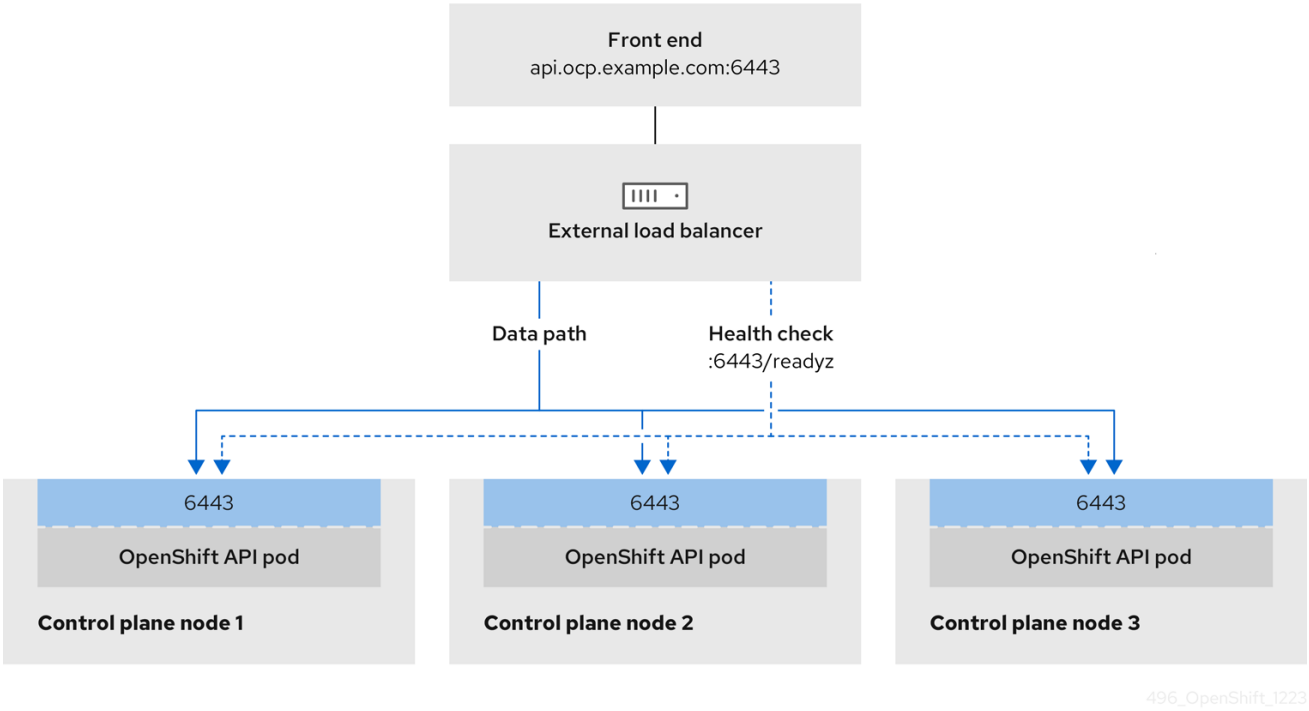
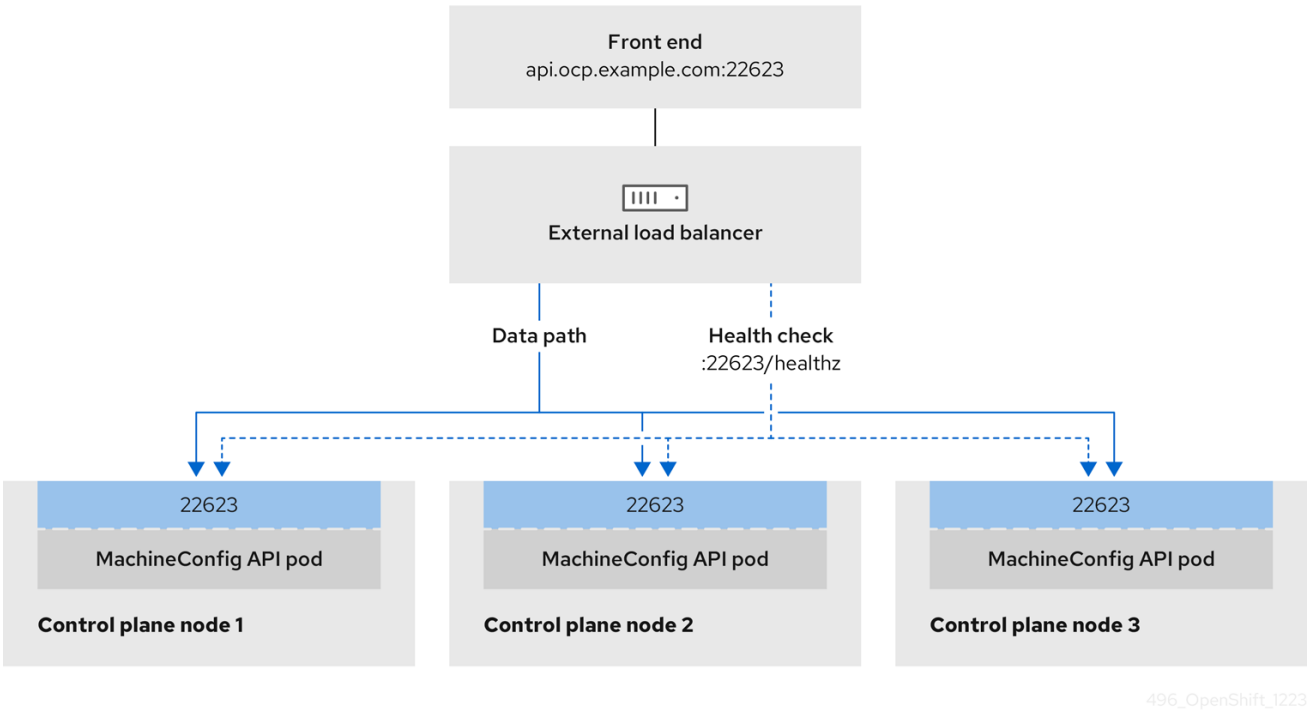


Figure 3.3. Example network workflow that shows an OpenShift MachineConfig API operating in an OpenShift Container Platform environment



The following configuration options are supported for user-managed load balancers:

- Use a node selector to map the Ingress Controller to a specific set of nodes. You must assign a static IP address to each node in this set, or configure each node to receive the same IP address from the Dynamic Host Configuration Protocol (DHCP). Infrastructure nodes commonly receive this type of configuration.

- Target all IP addresses on a subnet. This configuration can reduce maintenance overhead, because you can create and destroy nodes within those networks without reconfiguring the load balancer targets. If you deploy your ingress pods by using a machine set on a smaller network, such as a /**27** or /**28**, you can simplify your load balancer targets.

  TIP

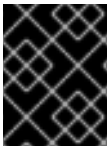
  You can list all IP addresses that exist in a network by checking the machine config pool's resources.

Before you configure a user-managed load balancer for your OpenShift Container Platform cluster, consider the following information:

- For a front-end IP address, you can use the same IP address for the front-end IP address, the Ingress Controller's load balancer, and API load balancer. Check the vendor's documentation for this capability.
- For a back-end IP address, ensure that an IP address for an OpenShift Container Platform control plane node does not change during the lifetime of the user-managed load balancer. You can achieve this by completing one of the following actions:
  - Assign a static IP address to each control plane node.
  - Configure each node to receive the same IP address from the DHCP every time the node requests a DHCP lease. Depending on the vendor, the DHCP lease might be in the form of an IP reservation or a static DHCP assignment.
- Manually define each node that runs the Ingress Controller in the user-managed load balancer for the Ingress Controller back-end service. For example, if the Ingress Controller moves to an undefined node, a connection outage can occur.

3.13.1. Configuring a user-managed load balancer

To integrate your infrastructure with existing network standards or gain more control over traffic management in OpenShift Container Platform , use a user-managed load balancer in place of the default load balancer.



IMPORTANT

Before you configure a user-managed load balancer, ensure that you read the "Services for a user-managed load balancer" section.

Read the following prerequisites that apply to the service that you want to configure for your user-managed load balancer.

NOTE

MetalLB, which runs on a cluster, functions as a user-managed load balancer.

Prerequisites

The following list details OpenShift API prerequisites:

- You defined a front-end IP address.

- TCP ports 6443 and 22623 are exposed on the front-end IP address of your load balancer. Check the following items:
  - Port 6443 provides access to the OpenShift API service.
  - Port 22623 can provide ignition startup configurations to nodes.
- The front-end IP address and port 6443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address and port 22623 are reachable only by OpenShift Container Platform nodes.
- The load balancer backend can communicate with OpenShift Container Platform control plane nodes on port 6443 and 22623.

The following list details Ingress Controller prerequisites:

- You defined a front-end IP address.
- TCP port 443 and port 80 are exposed on the front-end IP address of your load balancer.
- The front-end IP address, port 80 and port 443 are reachable by all users of your system with a location external to your OpenShift Container Platform cluster.
- The front-end IP address, port 80 and port 443 are reachable by all nodes that operate in your OpenShift Container Platform cluster.
- The load balancer backend can communicate with OpenShift Container Platform nodes that run the Ingress Controller on ports 80, 443, and 1936.

The following list details prerequisites for health check URL specifications:

You can configure most load balancers by setting health check URLs that determine if a service is available or unavailable. OpenShift Container Platform provides these health checks for the OpenShift API, Machine Configuration API, and Ingress Controller backend services.

The following example shows a Kubernetes API health check specification for a backend service:

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

The following example shows a Machine Config API health check specification for a backend service:

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

The following example shows a Ingress Controller health check specification for a backend service:

```
Path: HTTP:1936/healthz/ready
```

```
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

Procedure

1. Configure the HAProxy Ingress Controller, so that you can enable access to the cluster from your load balancer on ports 6443, 22623, 443, and 80. Depending on your needs, you can specify the IP address of a single subnet or IP addresses from multiple subnets in your HAProxy configuration.

   Example HAProxy configuration with one listed subnet

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
   bind 192.168.1.100:80
   mode tcp
   balance roundrobin
  option httpchk
```

```
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

Example HAProxy configuration with multiple listed subnets

```
# ...
listen api-server-6443
    bind *:6443
    mode tcp
      server master-00 192.168.83.89:6443 check inter 1s
      server master-01 192.168.84.90:6443 check inter 1s
      server master-02 192.168.85.99:6443 check inter 1s
      server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
    bind *:22623
    mode tcp
      server master-00 192.168.83.89:22623 check inter 1s
      server master-01 192.168.84.90:22623 check inter 1s
      server master-02 192.168.85.99:22623 check inter 1s
      server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
    bind *:80
    mode tcp
    balance source
      server worker-00 192.168.83.100:80 check inter 1s
      server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
    bind *:443
    mode tcp
    balance source
      server worker-00 192.168.83.100:443 check inter 1s
      server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
    bind *:6385
    mode tcp
    balance source
      server master-00 192.168.83.89:6385 check inter 1s
      server master-01 192.168.84.90:6385 check inter 1s
      server master-02 192.168.85.99:6385 check inter 1s
      server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
    bind *:5050
    mode tcp
    balance source
      server master-00 192.168.83.89:5050 check inter 1s
      server master-01 192.168.84.90:5050 check inter 1s
```

```
    server master-02 192.168.85.99:5050 check inter 1s
    server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. Use the **curl** CLI command to verify that the user-managed load balancer and its resources are operational:

   a. Verify that the cluster machine configuration API is accessible to the Kubernetes API server resource, by running the following command and observing the response:

   ```
   $ curl https://<loadbalancer_ip_address>:6443/version --insecure
   ```

   If the configuration is correct, you receive a JSON object in response:

   ```
   {
     "major": "1",
     "minor": "11+",
     "gitVersion": "v1.11.0+ad103ed",
     "gitCommit": "ad103ed",
     "gitTreeState": "clean",
     "buildDate": "2019-01-09T06:44:10Z",
     "goVersion": "go1.10.3",
     "compiler": "gc",
     "platform": "linux/amd64"
   }
   ```

   b. Verify that the cluster machine configuration API is accessible to the Machine config server resource, by running the following command and observing the output:

   ```
   $ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
   ```

   If the configuration is correct, the output from the command shows the following response:

   ```
   HTTP/1.1 200 OK
   Content-Length: 0
   ```

   c. Verify that the controller is accessible to the Ingress Controller resource on port 80, by running the following command and observing the output:

   ```
   $ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
   http://<load_balancer_front_end_IP_address>
   ```

   If the configuration is correct, the output from the command shows the following response:

   ```
   HTTP/1.1 302 Found
   content-length: 0
   location: https://console-openshift-console.apps.ocp4.private.opequon.net/
   cache-control: no-cache
   ```

   d. Verify that the controller is accessible to the Ingress Controller resource on port 443, by running the following command and observing the output:

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-
console.apps.<cluster_name>.<base_domain>
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

3. Configure the DNS records for your cluster to target the front-end IP addresses of the user-managed load balancer. You must update records to your DNS server for the cluster API and applications over the load balancer. The following examples shows modified DNS records:

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>  A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

IMPORTANT

DNS propagation might take some time for each DNS record to become available. Ensure that each DNS record propagates before validating each record.

4. For your OpenShift Container Platform cluster to use the user-managed load balancer, you must specify the following configuration in your cluster's **install-config.yaml** file:

```
# ...
platform:
  nutanix:
    loadBalancer:
      type: UserManaged
    apiVIPs:
    - <api_ip> 1
    ingressVIPs:
    - <ingress_ip> 2
# ...
```

where:

loadBalancer.type

Set **UserManaged** for the **type** parameter to specify a user-managed load balancer for your cluster. The parameter defaults to **OpenShiftManagedDefault**, which denotes the default internal load balancer. For services defined in an **openshift-kni-infra** namespace, a user-managed load balancer can deploy the **coredns** service to pods in your cluster but ignores **keepalived** and **haproxy** services.

loadBalancer.<api_ip>

Specifies a user-managed load balancer. Specify the user-managed load balancer's public IP address, so that the Kubernetes API can communicate with the user-managed load balancer. Mandatory parameter.

loadBalancer.<ingress_ip>

Specifies a user-managed load balancer. Specify the user-managed load balancer's public IP address, so that the user-managed load balancer can manage ingress traffic for your cluster. Mandatory parameter.

Verification

1. Use the **curl** CLI command to verify that the user-managed load balancer and DNS record configuration are operational:

   a. Verify that you can access the cluster API, by running the following command and observing the output:

   ```
   $ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
   ```

   If the configuration is correct, you receive a JSON object in response:

   ```
   {
     "major": "1",
     "minor": "11+",
     "gitVersion": "v1.11.0+ad103ed",
     "gitCommit": "ad103ed",
     "gitTreeState": "clean",
     "buildDate": "2019-01-09T06:44:10Z",
     "goVersion": "go1.10.3",
     "compiler": "gc",
     "platform": "linux/amd64"
     }
   ```

   b. Verify that you can access the cluster machine configuration, by running the following command and observing the output:

   ```
   $ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
   ```

   If the configuration is correct, the output from the command shows the following response:

   ```
   HTTP/1.1 200 OK
   Content-Length: 0
   ```

   c. Verify that you can access each cluster application on port 80, by running the following command and observing the output:

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --
insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

d. Verify that you can access each cluster application on port 443, by running the following command and observing the output:

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --
insecure
```

If the configuration is correct, the output from the command shows the following response:

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

3.14. 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 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
```

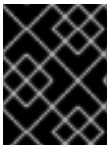
1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

1 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.

3.15. CONFIGURING THE DEFAULT STORAGE CONTAINER

After you install the cluster, you must install the Nutanix CSI Operator and configure the default storage container for the cluster.

For more information, see the Nutanix documentation for installing the CSI Operator and configuring registry storage.

3.16. 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.

3.17. ADDITIONAL RESOURCES

- About remote health monitoring

3.18. NEXT STEPS

- Remote health reporting
- Customize your cluster

CHAPTER 4. INSTALLING A CLUSTER ON NUTANIX IN A DISCONNECTED ENVIRONMENT

In OpenShift Container Platform 4.19, you can install a cluster on Nutanix infrastructure in a restricted network by creating an internal mirror of the installation release content.

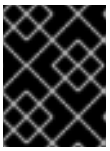
4.1. PREREQUISITES

- You have reviewed details about the OpenShift Container Platform installation and update processes.
- The installation program requires access to port 9440 on Prism Central and Prism Element. You verified that port 9440 is accessible.
- If you use a firewall, you have met these prerequisites:
  - You confirmed that port 9440 is accessible. Control plane nodes must be able to reach Prism Central and Prism Element on port 9440 for the installation to succeed.
  - You configured the firewall to grant access to the sites that OpenShift Container Platform requires. This includes the use of Telemetry.
- If your Nutanix environment is using the default self-signed SSL/TLS certificate, replace it with a certificate that is signed by a CA. The installation program requires a valid CA-signed certificate to access to the Prism Central API. For more information about replacing the self-signed certificate, see the Nutanix AOS Security Guide .
  If your Nutanix environment uses an internal CA to issue certificates, you must configure a cluster-wide proxy as part of the installation process. For more information, see Configuring a custom PKI.

IMPORTANT

Use 2048-bit certificates. The installation fails if you use 4096-bit certificates with Prism Central 2022.x.

- You have a container image registry, such as Red Hat Quay. If you do not already have a registry, you can create a mirror registry using *mirror registry for Red Hat OpenShift*.
- You have used the oc-mirror OpenShift CLI (oc) plugin to mirror all of the required OpenShift Container Platform content and other images, including the Nutanix CSI Operator, to your mirror registry.



IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

4.2. ABOUT INSTALLATIONS IN RESTRICTED NETWORKS

In OpenShift Container Platform 4.19, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

4.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

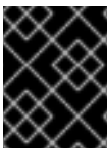
- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

4.3. 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.

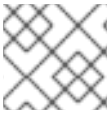
The SSH public key gets added to the **~/.ssh/authorized_keys** list for the **core** user on each node. After the key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access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

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

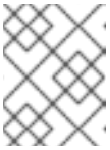
On some distributions, default SSH private key identities such as ~/**.ssh/id_rsa** and ~/**.ssh/id_dsa** are managed automatically.

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.

4.4. ADDING NUTANIX ROOT CA CERTIFICATES TO YOUR SYSTEM TRUST

Because the installation program requires access to the Prism Central API, you must add your Nutanix trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the Prism Central web console, download the Nutanix root CA certificates.
2. Extract the compressed file that contains the Nutanix root CA certificates.
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

4.5. DOWNLOADING THE RHCOS CLUSTER IMAGE

Prism Central requires access to the Red Hat Enterprise Linux CoreOS (RHCOS) image to install the cluster. You can use the installation program to locate and download the RHCOS image and make it available through an internal HTTP server or Nutanix Objects.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

Procedure

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

```
$ ./openshift-install coreos print-stream-json
```

2. Use the output of the command to find the location of the Nutanix image, and click the link to download it.

Example output

```
"nutanix": {
  "release": "411.86.202210041459-0",
  "formats": {
    "qcow2": {
      "disk": {
```

```
        "location": "https://rhcos.mirror.openshift.com/art/storage/releases/rhcos-
4.11/411.86.202210041459-0/x86_64/rhcos-411.86.202210041459-0-
nutanix.x86_64.qcow2",
        "sha256":
"42e227cac6f11ac37ee8a2f9528bb3665146566890577fd55f9b950949e5a54b"
```

3. Make the image available through an internal HTTP server or Nutanix Objects.

4. Note the location of the downloaded image. You update the **platform** section in the installation configuration file (**install-config.yaml**) with the image's location before deploying the cluster.

Snippet of an install-config.yaml file that specifies the RHCOS image

```
platform:
  nutanix:
    clusterOSImage: http://example.com/images/rhcos-411.86.202210041459-0-
nutanix.x86_64.qcow2
```

4.6. CREATING THE INSTALLATION CONFIGURATION FILE

You can customize the OpenShift Container Platform cluster you install on

Nutanix.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- You have the **imageContentSourcePolicy.yaml** file that was created when you mirrored your registry.
- You have the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image you download.
- You have obtained the contents of the certificate for your mirror registry.
- You have retrieved a Red Hat Enterprise Linux CoreOS (RHCOS) image and uploaded it to an accessible location.
- You have verified that you have met the Nutanix networking requirements. For more information, see "Preparing to install on Nutanix".

Procedure

1. Create the **install-config.yaml** file.

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

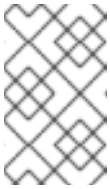
   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   - **<installation_directory>**: For **<installation_directory>**, specify the directory name to store the files that the installation program creates.
     When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore 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.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access 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.

ii. Select **nutanix** as the platform to target.

iii. Enter the Prism Central domain name or IP address.

iv. Enter the port that is used to log into Prism Central.

v. Enter the credentials that are used to log into Prism Central.
The installation program connects to Prism Central.

vi. Select the Prism Element that will manage the OpenShift Container Platform cluster.

vii. Select the network subnet to use.

viii. Enter the virtual IP address that you configured for control plane API access.

ix. Enter the virtual IP address that you configured for cluster ingress.

x. Enter the base domain. This base domain must be the same one that you configured in the DNS records.

xi. Enter a descriptive name for your cluster.
The cluster name you enter must match the cluster name you specified when configuring the DNS records.

2. In the **install-config.yaml** file, set the value of **platform.nutanix.clusterOSImage** to the image location or name. For example:

```
platform:
  nutanix:
    clusterOSImage: http://mirror.example.com/images/rhcos-47.83.202103221318-0-
nutanix.x86_64.qcow2
```

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.

a. Update the **pullSecret** value to contain the authentication information for your registry:

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

b. Add the **additionalTrustBundle** parameter and value.

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

c. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

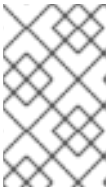
For these values, use the **imageContentSourcePolicy.yaml** file that was created when you mirrored the registry.

d. Optionally, set the publishing strategy to **Internal**:

```
publish: Internal
```

By setting this option, you create an internal Ingress Controller and a private load balancer.

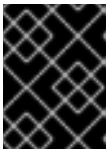
4. Optional: Update one or more of the default configuration parameters in the **install.config.yaml** file to customize the installation.
   For more information about the parameters, see "Installation configuration parameters".



NOTE

If you are installing a three-node cluster, be sure to set the **compute.replicas** parameter to **0**. This ensures that cluster's control planes are schedulable. For more information, see "Installing a three-node cluster on {platform}".

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



IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- Installation configuration parameters for Nutanix

4.6.1. Sample customized install-config.yaml file for Nutanix

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. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    nutanix: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
      categories: (5)
      - key: <category_key_name>
        value: <category_value>
controlPlane: (6)
  hyperthreading: Enabled (7)
  name: master
  replicas: 3
  platform:
    nutanix: (8)
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
      osDisk:
        diskSizeGiB: 120
      categories: (9)
      - key: <category_key_name>
        value: <category_value>
metadata:
  creationTimestamp: null
  name: test-cluster (10)
networking:
```

```
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  nutanix:
    apiVIP: 10.40.142.7 12
    ingressVIP: 10.40.142.8 13
    defaultMachinePlatform:
      bootType: Legacy
      categories: 14
      - key: <category_key_name>
        value: <category_value>
      project: 15
        type: name
        name: <project_name>
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 16
        port: 9440 17
      password: <password> 18
      username: <username> 19
    prismElements:
    - endpoint:
        address: your.prismelement.domainname
        port: 9440
      uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
    subnetUUIDs:
    - c7938dc6-7659-453e-a688-e26020c68e43
    clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
20
credentialsMode: Manual
publish: External
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
additionalTrustBundle: | 24
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 25
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 10 12 13 16 17 18 19 Required. The installation program prompts you for this value.

2 6 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

3 7 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4 8 Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

5 9 14 Optional: Provide one or more pairs of a prism category key and a prism category value. These category key-value pairs must exist in Prism Central. You can provide separate categories to compute machines, control plane machines, or all machines.

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

15 Optional: Specify a project with which VMs are associated. Specify either **name** or **uuid** for the project type, and then provide the corresponding UUID or project name. You can associate projects to compute machines, control plane machines, or all machines.

20 Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server or Nutanix Objects and pointing the installation program to the image.

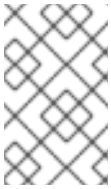
21 For **<local_registry>**, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example **registry.example.com** or **registry.example.com:5000**. For **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

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 Optional: You can 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 Provide the contents of the certificate file that you used for your mirror registry.

25 Provide these values from the **metadata.name: release-0** section of the **imageContentSourcePolicy.yaml** file that was created when you mirrored the registry.

4.6.2. Configuring failure domains

Failure domains improve the fault tolerance of an OpenShift Container Platform cluster by distributing control plane and compute machines across multiple Nutanix Prism Elements (clusters).

TIP

It is recommended that you configure three failure domains to ensure high-availability.

Prerequisites

- You have an installation configuration file (**install-config.yaml**).

Procedure

1. Edit the **install-config.yaml** file and add the following stanza to configure the first failure domain:

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
platform:
  nutanix:
    failureDomains:
    - name: <failure_domain_name>
      prismElement:
        name: <prism_element_name>
        uuid: <prism_element_uuid>
      subnetUUIDs:
      - <network_uuid>
# ...
```

where:

<failure_domain_name>

Specifies a unique name for the failure domain. The name is limited to 64 or fewer characters, which can include lower-case letters, digits, and a dash (**-**). The dash cannot be in the leading or ending position of the name.

<prism_element_name>

Optional. Specifies the name of the Prism Element.

<prism_element_uuid>

Specifies the UUID of the Prism Element.

<network_uuid>

Specifies the one or more UUIDs of the Prism Element subnet objects. Among them, one of the subnet's IP address prefixes (CIDRs) must contain the virtual IP addresses that the OpenShift Container Platform cluster uses. A maximum of 32 subnets for each failure domain (Prism Element) in an OpenShift Container Platform cluster is supported. All **subnetUUID** values must be unique.

2. As required, configure additional failure domains.

3. To distribute control plane and compute machines across the failure domains, do one of the following:

   - If compute and control plane machines can share the same set of failure domains, add the failure domain names under the cluster's default machine configuration.

     Example of control plane and compute machines sharing a set of failure domains

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     platform:
       nutanix:
         defaultMachinePlatform:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     ```

   - If compute and control plane machines must use different failure domains, add the failure domain names under the respective machine pools.

     Example of control plane and compute machines using different failure domains

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     controlPlane:
       platform:
         nutanix:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     compute:
       platform:
         nutanix:
           failureDomains:
     ```

```
        - failure-domain-1
        - failure-domain-2
# ...
```

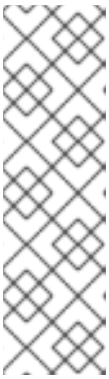
4. Save the file.

4.6.3. 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

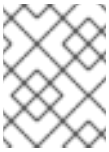
Specifies a comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use ***** to bypass the proxy for all destinations.

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.

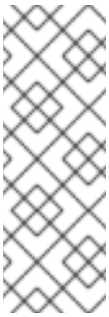
additionalTrustBundlePolicy

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.

4.7. 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.

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 v4.19 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>
```

4.8. 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 v4.19 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
```

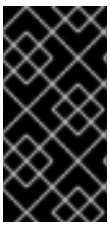
Verification

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

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

4.9. 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.



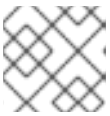
IMPORTANT

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 v4.19 macOS Clients** entry and save the file.



NOTE

For macOS arm64, choose the **OpenShift v4.19 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>
```

4.10. CONFIGURING IAM FOR NUTANIX

Installing the cluster requires that the Cloud Credential Operator (CCO) operate in manual mode. While the installation program configures the CCO for manual mode, you must specify the identity and access management secrets.

Prerequisites

- You have configured the **ccoctl** binary.
- You have an **install-config.yaml** file.

Procedure

1. Create a YAML file that contains the credentials data in the following format:

   #### Credentials data format

   ```
   credentials:
   - type: basic_auth (1)
     data:
       prismCentral: (2)
         username: <username_for_prism_central>
         password: <password_for_prism_central>
       prismElements: (3)
       - name: <name_of_prism_element>
         username: <username_for_prism_element>
         password: <password_for_prism_element>
   ```

   (1) Specify the authentication type. Only basic authentication is supported.

   (2) Specify the Prism Central credentials.

   (3) Optional: Specify the Prism Element credentials.

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.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
  secretRef:
    name: nutanix-credentials
    namespace: openshift-machine-api
```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

```
$ ccoctl nutanix create-shared-secrets \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --credentials-source-filepath=<path_to_credentials_file> 3
```

1. Specify the path to the directory that contains the files for the component **CredentialsRequests** objects.
2. Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.
3. Optional: Specify the directory that contains the credentials data YAML file. By default, **ccoctl** expects this file to be in **<home_directory>/.nutanix/credentials**.

5. Edit the **install-config.yaml** configuration file so that the **credentialsMode** parameter is set to **Manual**.

Example install-config.yaml configuration file

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
```

```
...
```

1 Add this line to set the **credentialsMode** parameter to **Manual**.

6. Create the installation manifests by running the following command:

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

1 Specify the path to the directory that contains the **install-config.yaml** file for your cluster.

7. Copy the generated credential files to the target manifests directory by running the following command:

```
$ cp <ccoctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

Verification

- Ensure that the appropriate secrets exist in the **manifests** directory.

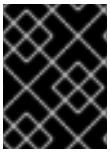
```
$ ls ./<installation_directory>/manifests
```

Example output

```
cluster-config.yaml
cluster-dns-02-config.yml
cluster-infrastructure-02-config.yml
cluster-ingress-02-config.yml
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-proxy-01-config.yaml
cluster-scheduler-02-config.yml
cvo-overrides.yaml
kube-cloud-config.yaml
kube-system-configmap-root-ca.yaml
machine-config-server-tls-secret.yaml
openshift-config-secret-pull-secret.yaml
openshift-cloud-controller-manager-nutanix-credentials-credentials.yaml
openshift-machine-api-nutanix-credentials-credentials.yaml
```

4.11. 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 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
```

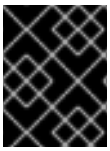
1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

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**.

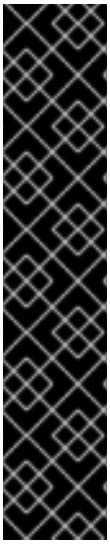


IMPORTANT

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.

4.12. POST INSTALLATION

Complete the following steps to complete the configuration of your cluster.

4.12.1. Disabling the default OperatorHub catalog sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding **disableAllDefaultSources: true** to the **OperatorHub** object:

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

TIP

Alternatively, you can use the web console to manage catalog sources. From the **Administration → Cluster Settings → Configuration → OperatorHub** page, click the **Sources** tab, where you can create, update, delete, disable, and enable individual sources.

4.12.2. Installing the policy resources into the cluster

Mirroring the OpenShift Container Platform content using the oc-mirror OpenShift CLI (oc) plugin creates resources, which include **catalogSource-certified-operator-index.yaml** and **imageContentSourcePolicy.yaml**.

- The **ImageContentSourcePolicy** resource associates the mirror registry with the source registry and redirects image pull requests from the online registries to the mirror registry.
- The **CatalogSource** resource is used by Operator Lifecycle Manager (OLM) Classic to retrieve information about the available Operators in the mirror registry, which lets users discover and install Operators.

NOTE

OLM v1 uses the **ClusterCatalog** resource to retrieve information about the available cluster extensions in the mirror registry.

The oc-mirror plugin v1 does not generate **ClusterCatalog** resources automatically; you must manually create them. The oc-mirror plugin v2 does, however, generate **ClusterCatalog** resources automatically.

For more information on creating and applying **ClusterCatalog** resources, see "Adding a catalog to a cluster" in "Extensions".

After you install the cluster, you must install these resources into the cluster.

Prerequisites

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

Procedure

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

```
$ oc apply -f ./oc-mirror-workspace/results-<id>/
```

Verification

1. Verify that the **ImageContentSourcePolicy** resources were successfully installed:

```
$ oc get imagecontentsourcepolicy
```

2. Verify that the **CatalogSource** resources were successfully installed:

```
$ oc get catalogsource --all-namespaces
```

Additional resources

- Adding a catalog to a cluster in "Extensions"

4.12.3. Configuring the default storage container

After you install the cluster, you must install the Nutanix CSI Operator and configure the default storage container for the cluster.

For more information, see the Nutanix documentation for installing the CSI Operator and configuring registry storage.

4.13. 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.

4.14. ADDITIONAL RESOURCES

- About remote health monitoring

4.15. NEXT STEPS

- If necessary, see Remote health reporting
- If necessary, see Registering your disconnected cluster
- Customize your cluster

CHAPTER 5. INSTALLING A THREE-NODE CLUSTER ON NUTANIX

In OpenShift Container Platform version 4.19, you can install a three-node cluster on Nutanix. A three-node cluster consists of three control plane machines, which also act as compute machines. This type of cluster provides a smaller, more resource efficient cluster, for cluster administrators and developers to use for testing, development, and production.

5.1. CONFIGURING A THREE-NODE CLUSTER

To configure a three-node cluster, set the number of worker nodes to **0** in the **install-config.yaml** file before you deploy the cluster.

Setting the number of worker nodes to **0** ensures that the control plane machines are schedulable. This allows application workloads to be scheduled to run from the control plane nodes.

NOTE

Because application workloads run from control plane nodes, additional subscriptions are required, as the control plane nodes are considered to be compute nodes.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

- Set the number of compute replicas to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

  Example install-config.yaml file for a three-node cluster

```
apiVersion: v1
baseDomain: example.com
compute:
- name: worker
  platform: {}
  replicas: 0
# ...
```

5.2. NEXT STEPS

- Installing a cluster on Nutanix

CHAPTER 6. UNINSTALLING A CLUSTER ON NUTANIX

You can remove a cluster that you deployed to Nutanix.

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 that were not removed properly, especially with user-provisioned infrastructure clusters. Some resources might exist because either the installation program did not create the resource or could not access the resource.

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

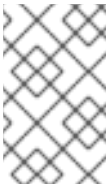
1. From the directory that has the installation program on the computer that you used to install the cluster, run the following command:

```
$ ./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 includes 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.

CHAPTER 7. INSTALLATION CONFIGURATION PARAMETERS FOR NUTANIX

Before you deploy an OpenShift Container Platform cluster on Nutanix, you provide parameters to customize your cluster and the platform that hosts it. When you create the **install-config.yaml** file, you provide values for the required parameters through the command line. You can then modify the **install-config.yaml** file to customize your cluster further.

7.1. AVAILABLE INSTALLATION CONFIGURATION PARAMETERS FOR NUTANIX

The following tables specify the required, optional, and Nutanix-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.

7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.1. Required parameters

Parameter	Description
apiVersion:	The API version for the **install-config.yaml** content. The current version is **v1**. The installation program might also support older API versions. **Value:** 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. **Value:** A fully-qualified domain or subdomain name, such as **example.com**.
metadata:	Kubernetes resource **ObjectMeta**, from which only the **name** parameter is consumed. **Value:** Object

Parameter	Description
`metadata:` `  name:`	The name of the cluster. DNS records for the cluster are all subdomains of **{{.metadata.name}}.{{.baseDomain}}**. **Value:** String of lowercase letters and hyphens (-), 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. **Value:** Object

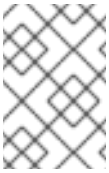
`pullSecret:`	Get a pull secret from Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.
	
	Value:
	
	`{`
	`  "auths":{`
	`  "cloud.openshift.com":{`
	`  "auth":"b3Blb=",`
	`  "email":"you@example.com"`
	`  },`
	`  "quay.io":{`
	`  "auth":"b3Blb=",`
	`  "email":"you@example.com"`
	`  }`
	`  }`
	`}`

7.1.2. 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 7.2. Network parameters

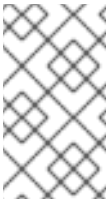
Parameter	Description
networking:	The configuration for the cluster network. **Value:** Object  **NOTE** You cannot change parameters specified by the **networking** object after installation.
networking: networkType:	The Red Hat OpenShift Networking network plugin to install. Value:**OVNKubernetes**. **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. **Value:** 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. **Value:** An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between **0** and **32**.

Parameter	Description

`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.
	
	Value: A subnet prefix.
	
	The default value is **23**.

`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.
	
	Value: 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.
	
	Value: An array of objects. For example:
	
	`networking:`
	`  machineNetwork:`
	`  - cidr: 10.0.0.0/16`

Parameter	Description
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**. **Value:** 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.
networking: ovnKubernetesConfig: ipv4: internalJoinSubnet:	Configures the IPv4 join subnet that is used internally by **ovn-kubernetes**. This subnet must not overlap with any other subnet that OpenShift Container Platform is using, including the node network. The size of the subnet must be larger than the number of nodes. You cannot change the value after installation. **Value:** An IP network block in CIDR notation. The default value is **100.64.0.0/16**.

7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 7.3. Optional parameters

Parameter	Description
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. **Value:** String

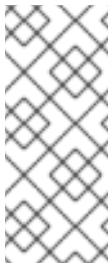
Parameter	Description
capabilities:	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in *Installing*. **Value:** String array
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**. **Value:** String
capabilities: additionalEnabledCapabilities:	Extends the set of optional capabilities beyond what you specify in **baselineCapabilitySet**. You can specify multiple capabilities in this parameter. **Value:** 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. **Value:** **None** or **AllNodes**. **None** is the default value.
compute:	The configuration for the machines that comprise the compute nodes. **Value:** 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). **Value:** String

Parameter	Description
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. Value: **Enabled** or **Disabled**
compute: name:	Required if you use **compute**. The name of the machine pool. Value: **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. 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. **Value:** 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". **Value:** String. The name of the feature set to enable, such as **TechPreviewNoUpgrade**.
controlPlane:	The configuration for the machines that form the control plane. **Value:** Array of **MachinePool** objects.

Parameter	Description
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). **Value:** 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. Value: **Enabled** or **Disabled**
controlPlane: name:	Required if you use **controlPlane**. The name of the machine pool. Value: **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. Value:**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**
controlPlane: replicas:	The number of control plane machines to provision. **Value:** Supported values are **3**, or **1** when deploying single-node OpenShift.

Parameter	Description
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. Value: **Mint**, **Passthrough**, **Manual** or an empty string (**""**).

Parameter	Description
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.  **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. Value: **false** or **true**
imageContentSources:	Sources and repositories for the release-image content. **Value:** 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. **Value:** String

Parameter	Description
imageContentSources: mirrors:	Specify one or more repositories that might also contain the same images. **Value:** Array of strings
publish:	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes. Value:**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. **Value:** For example, **sshKey: ssh-ed25519 AAAA..**.

7.1.4. Additional Nutanix configuration parameters

Additional Nutanix configuration parameters are described in the following table:

Table 7.4. Additional Nutanix cluster parameters

Parameter	Description

`compute:`
`  platform:`
`  nutanix:`
`  categories:`
`  key:`
	The name of a prism category key to apply to compute VMs. This parameter must be accompanied by the **value** parameter, and both **key** and **value** parameters must exist in Prism Central. For more information on categories, see Category management.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  categories:`
`  value:`
	The value of a prism category key-value pair to apply to compute VMs. This parameter must be accompanied by the **key** parameter, and both **key** and **value** parameters must exist in Prism Central.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  failureDomains:`
	The failure domains that apply to only compute machines.
	
	Failure domains are specified in **platform.nutanix.failureDomains**.
	
	Value: List.
	
	The name of one or more failures domains.

`compute:`
`  platform:`
`  nutanix:`
`  gpus:`
`  type:`
	The type of identifier used to attach a GPU to a compute machine. Valid values are "Name" or "DeviceID".
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  gpus:`
`  name:`
	The name of the GPU device to attach to a compute machine. This parameter is required if the GPU **type** is "Name".
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  gpus:`
`  deviceID:`
	The device identifier of the GPU device to attach to a compute machine. This information is available in Prism Central. This parameter is required if the GPU **type** is "DeviceID".
	
	Value: Integer

Parameter	Description

`compute:`
`  platform:`
`  nutanix:`
`  project:`
`  type:`
	The type of identifier you use to select a project for compute VMs. Projects define logical groups of user roles for managing permissions, networks, and other parameters. For more information on projects, see Projects Overview.
	
	Value: **name** or **uuid**

`compute:`
`  platform:`
`  nutanix:`
`  project:`
`  name: or uuid:`
	The name or UUID of a project with which compute VMs are associated. This parameter must be accompanied by the **type** parameter.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  bootType:`
	The boot type that the compute machines use. You must use the **Legacy** boot type in OpenShift Container Platform 4.19. For more information on boot types, see Understanding UEFI, Secure Boot, and TPM in the Virtualized Environment.
	
	Value: **Legacy**, **SecureBoot** or **UEFI**. The default is **Legacy**.

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  dataSourceImage:`
`  name:`
	Optional. The name of the data source image for the virtual machine disk in Prism Central.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  dataSourceImage:`
`  referenceName:`
	Optional. The reference name of the data source image in the failure domain. If you use this parameter, you must configure a matching **dataSourceImage** with the same **referenceName** in each failure domain that the compute nodes occupy. For more information about configuring failure domains, see *Configuring failure domains* in the *Installing a cluster on Nutanix* page.
	
	Value: String

Parameter	Description

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  dataSourceImage:`
`  uuid:`
	The UUID of the data source image in Prism Central. This value is required.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  deviceProperties:`
`  adapterType:`
	The adapter type of the disk address. If the disk type is "Disk", valid values are "SCSI", "IDE", "PCI", "SATA" or "SPAPR". If the disk type is "CDRom", valid values are "IDE" or "SATA".
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  deviceProperties:`
`  deviceIndex:`
	The index of the disk address. Valid values are non-negative integers including **0**. The device index for disks that share the same adapter type should start at 0 and increase consecutively. The default value is **0**. For each virtual machine, the **Disk.SCSI.0** and **CDRom.IDE.0** indices are reserved. If you use the **Disk.SCSI** or **CDRom.IDE** disk and adapter types, the **deviceIndex** should start at **1**.
	
	Value: Non-negative integer, including **0**.

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  deviceProperties:`
`  deviceType:`
	The disk device type. Valid values are "Disk" and "CDRom".
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  diskSize:`
	The size of the disk to attach to the virtual machine. The minimum size is 1Gb.
	
	Value: Quantity format, such as 100G or 100Gi. For more information on this format, see link:https://pkg.go.dev/k8s.io/apimachinery/pkg/api/resource#Format.

Parameter	Description

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  storageConfig:`
`  diskMode:`
	The disk mode. Valid values are **Standard** or **Flash**, and the default is **Standard**.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  storageConfig:`
`  storageContainer:`
`  name:`
	Optional. The name of the storage container object used by the virtual machine disk in Prism Central.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  storageConfig:`
`  storageContainer:`
`  referenceName:`
	Optional. The reference name of the storage container in the failure domain. If you use this, you must configure a matching **storageContainer** with the same **referenceName** in each failure domain the compute nodes occupy. For more information about configuring failure domains, see *Configuring failure domains* in the *Installing a cluster on Nutanix* page.
	
	Value: String

`compute:`
`  platform:`
`  nutanix:`
`  dataDisks:`
`  storageConfig:`
`  storageContainer:`
`  uuid:`
	The UUID of the storage container in Prism Central.
	
	Value: String

`controlPlane:`
`  platform:`
`  nutanix:`
`  categories:`
`  key:`
	The name of a prism category key to apply to control plane VMs. This parameter must be accompanied by the **value** parameter, and both **key** and **value** parameters must exist in Prism Central. For more information on categories, see Category management.
	
	Value: String

Parameter	Description

`controlPlane:`
`  platform:`
`  nutanix:`
`  categories:`
`  value:`
	The value of a prism category key-value pair to apply to control plane VMs. This parameter must be accompanied by the **key** parameter, and both **key** and **value** parameters must exist in Prism Central.
	
	Value: String

`controlPlane:`
`  platform:`
`  nutanix:`
`  failureDomains:`
	The failure domains that apply to only control plane machines.
	
	Failure domains are specified in **platform.nutanix.failureDomains**.
	
	Value: List.
	
	The name of one or more failures domains.

`controlPlane:`
`  platform:`
`  nutanix:`
`  project:`
`  type:`
	The type of identifier you use to select a project for control plane VMs. Projects define logical groups of user roles for managing permissions, networks, and other parameters. For more information on projects, see Projects Overview.
	
	Value: name or **uuid**

`controlPlane:`
`  platform:`
`  nutanix:`
`  project:`
`  name: or uuid:`
	The name or UUID of a project with which control plane VMs are associated. This parameter must be accompanied by the **type** parameter.
	
	Value: String

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  categories:`
`  key:`
	The name of a prism category key to apply to all VMs. This parameter must be accompanied by the **value** parameter, and both **key** and **value** parameters must exist in Prism Central. For more information on categories, see Category management.
	
	Value: String

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  categories:`
`  value:`
	The value of a prism category key-value pair to apply to all VMs. This parameter must be accompanied by the **key** parameter, and both **key** and **value** parameters must exist in Prism Central.
	
	Value: String

Parameter	Description

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  failureDomains:`
	The failure domains that apply to both control plane and compute machines.
	
	Failure domains are specified in **platform.nutanix.failureDomains**.
	
	Value: List.
	
	The name of one or more failures domains.

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  project:`
`  type:`
	The type of identifier you use to select a project for all VMs. Projects define logical groups of user roles for managing permissions, networks, and other parameters. For more information on projects, see Projects Overview.
	
	Value: name or **uuid**.

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  project:`
`  name: or uuid:`
	The name or UUID of a project with which all VMs are associated. This parameter must be accompanied by the **type** parameter.
	
	Value: String

`platform:`
`  nutanix:`
`  defaultMachinePlatform:`
`  bootType:`
	The boot type for all machines. You must use the **Legacy** boot type in OpenShift Container Platform 4.19. For more information on boot types, see Understanding UEFI, Secure Boot, and TPM in the Virtualized Environment.
	
	Value: Legacy, **SecureBoot** or **UEFI**. The default is **Legacy**.

`platform:`
`  nutanix:`
`  apiVIP:`
	The virtual IP (VIP) address that you configured for control plane API access.
	
	Value: IP address

Parameter	Description

`platform:`
`  nutanix:`
`  failureDomains:`
`  - name:`
`  prismElement:`
`  name:`
`  uuid:`
`  subnetUUIDs:`
`  -`
	By default, the installation program installs cluster machines to a single Prism Element instance. A maximum of 32 subnets for each failure domain (Prism Element) in an OpenShift Container Platform cluster is supported. All **subnetUUID** values must be unique. You can specify additional Prism Element instances for fault tolerance, and then apply them to:
	● The cluster's default machine configuration
	● Only control plane or compute machine pools
	Value: A list of configured failure domains.
	For more information on usage, see "Configuring a failure domain" in "Installing a cluster on Nutanix".

`platform:`
`  nutanix:`
`  ingressVIP:`
	The virtual IP (VIP) address that you configured for cluster ingress.
	Value: IP address

`platform:`
`  nutanix:`
`  prismCentral:`
`  endpoint:`
`  address:`
	The Prism Central domain name or IP address.
	Value: String

`platform:`
`  nutanix:`
`  prismCentral:`
`  endpoint:`
`  port:`
	The port that is used to log into Prism Central.
	Value: String

`platform:`
`  nutanix:`
`  prismCentral:`
`  password:`
	The password for the Prism Central user name.
	Value: String

Parameter	Description

`platform:`
`  nutanix:`
`  preloadedOSImageName:`
	Instead of creating and uploading a RHCOS image object for each OpenShift Container Platform cluster, this parameter uses the named, preloaded RHCOS image object from the Prism Elements to which the OpenShift Container Platform cluster is deployed.
	
	Value: String

`platform:`
`  nutanix:`
`  prismCentral:`
`  username:`
	The user name that is used to log into Prism Central.
	
	Value: String

`platform:`
`  nutanix:`
`  prismElements:`
`  endpoint:`
`  address:`
	The Prism Element domain name or IP address. [1]
	
	Value: String

`platform:`
`  nutanix:`
`  prismElements:`
`  endpoint:`
`  port:`
	The port that is used to log into Prism Element.
	
	Value: String

`platform:`
`  nutanix:`
`  prismElements:`
`  uuid:`
	The universally unique identifier (UUID) for Prism Element.
	
	Value: String

`platform:`
`  nutanix:`
`  subnetUUIDs:`
	The UUID of the Prism Element network that contains the virtual IP addresses and DNS records that you configured. [2]
	
	Value: String

Parameter	Description

platform:
  nutanix:
  clusterOSImage:
	Optional: By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image. If Prism Central does not have internet access, you can override the default behavior by hosting the RHCOS image on any HTTP server and pointing the installation program to the image.
	
	Value: An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2

1. The **prismElements** section holds a list of Prism Elements (clusters). A Prism Element encompasses all of the Nutanix resources, for example virtual machines and subnets, that are used to host the OpenShift Container Platform cluster.

2. A maximum of 32 subnets for each Prism Element in an OpenShift Container Platform cluster is supported. All **subnetUUID** values must be unique.