OpenShift Container Platform 4.15

Installing on OpenStack

Installing OpenShift Container Platform on OpenStack

Last Updated: 2026-04-01

OpenShift Container Platform 4.15 Installing on OpenStack

Installing OpenShift Container Platform on OpenStack

Legal Notice

Abstract

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

Table of Contents

CHAPTER 1. PREPARING TO INSTALL ON OPENSTACK

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP).

1.1. PREREQUISITES

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

1.2. CHOOSING A METHOD TO INSTALL OPENSHIFT CONTAINER PLATFORM ON OPENSTACK

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

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

1.2.1. Installing a cluster on installer-provisioned infrastructure

You can install a cluster on Red Hat OpenStack Platform (RHOSP) infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- **Installing a cluster on OpenStack with customizations**: You can install a customized cluster on RHOSP. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available post-installation.
- **Installing a cluster on OpenStack in a restricted network**: You can install OpenShift Container Platform on RHOSP in a restricted or disconnected network by creating an internal mirror of the installation release content. You can use this method to install a cluster that does not require an active internet connection to obtain the software components. You can also use this installation method to ensure that your clusters only use container images that satisfy your organizational controls on external content.

1.2.2. Installing a cluster on user-provisioned infrastructure

You can install a cluster on RHOSP infrastructure that you provision, by using one of the following methods:

- **Installing a cluster on OpenStack on your own infrastructure**: You can install OpenShift Container Platform on user-provisioned RHOSP infrastructure. By using this installation method, you can integrate your cluster with existing infrastructure and modifications. For installations on user-provisioned infrastructure, you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. You can use the provided Ansible playbooks to assist with the deployment process.

1.3. SCANNING RHOSP ENDPOINTS FOR LEGACY HTTPS CERTIFICATES

Beginning with OpenShift Container Platform 4.10, HTTPS certificates must contain subject alternative name (SAN) fields. Run the following script to scan each HTTPS endpoint in a Red Hat OpenStack Platform (RHOSP) catalog for legacy certificates that only contain the **CommonName** field.

IMPORTANT

OpenShift Container Platform does not check the underlying RHOSP infrastructure for legacy certificates prior to installation or updates. Use the provided script to check for these certificates yourself. Failing to update legacy certificates prior to installing or updating a cluster will result in cluster dysfunction.

Prerequisites

- On the machine where you run the script, have the following software:
  - Bash version 4.0 or greater
  - **grep**
  - OpenStack client
  - **jq**
  - OpenSSL version 1.1.1l or greater
- Populate the machine with RHOSP credentials for the target cloud.

Procedure

1. Save the following script to your machine:

```
#!/usr/bin/env bash

set -Eeuo pipefail

declare catalog san
catalog="$(mktemp)"
san="$(mktemp)"
readonly catalog san

declare invalid=0

openstack catalog list --format json --column Name --column Endpoints \
 | jq -r '.[] | .Name as $name | .Endpoints[] | select(.interface=="public") | [$name, .interface,
.url] | join(" ")' \
 | sort \
 > "$catalog"

while read -r name interface url; do
 # Ignore HTTP
 if [[ ${url#"http://"} != "$url" ]]; then
  continue
```

```
 fi

 # Remove the schema from the URL
 noschema=${url#"https://"}

 # If the schema was not HTTPS, error
 if [[ "$noschema" == "$url" ]]; then
  echo "ERROR (unknown schema): $name $interface $url"
  exit 2
 fi

 # Remove the path and only keep host and port
 noschema="${noschema%%/*}"
 host="${noschema%%:*}"
 port="${noschema##*:}"

 # Add the port if was implicit
 if [[ "$port" == "$host" ]]; then
  port='443'
 fi

 # Get the SAN fields
 openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
  | openssl x509 -noout -ext subjectAltName \
  > "$san"

 # openssl returns the empty string if no SAN is found.
 # If a SAN is found, openssl is expected to return something like:
 #
 #    X509v3 Subject Alternative Name:
 #        DNS:standalone, DNS:osp1, IP Address:192.168.2.1, IP Address:10.254.1.2
 if [[ "$(grep -c "Subject Alternative Name" "$san" || true)" -gt 0 ]]; then
  echo "PASS: $name $interface $url"
 else
  invalid=$((invalid+1))
  echo "INVALID: $name $interface $url"
 fi
done < "$catalog"

# clean up temporary files
rm "$catalog" "$san"

if [[ $invalid -gt 0 ]]; then
 echo "${invalid} legacy certificates were detected. Update your certificates to include a SAN
field."
 exit 1
else
 echo "All HTTPS certificates for this cloud are valid."
fi
```

2. Run the script.

3. Replace any certificates that the script reports as **INVALID** with certificates that contain SAN fields.

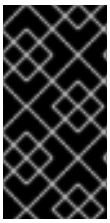
IMPORTANT

You must replace all legacy HTTPS certificates before you install OpenShift Container Platform 4.10 or update a cluster to that version. Legacy certificates will be rejected with the following message:

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

1.3.1. Scanning RHOSP endpoints for legacy HTTPS certificates manually

Beginning with OpenShift Container Platform 4.10, HTTPS certificates must contain subject alternative name (SAN) fields. If you do not have access to the prerequisite tools that are listed in "Scanning RHOSP endpoints for legacy HTTPS certificates", perform the following steps to scan each HTTPS endpoint in a Red Hat OpenStack Platform (RHOSP) catalog for legacy certificates that only contain the **CommonName** field.



IMPORTANT

OpenShift Container Platform does not check the underlying RHOSP infrastructure for legacy certificates prior to installation or updates. Use the following steps to check for these certificates yourself. Failing to update legacy certificates prior to installing or updating a cluster will result in cluster dysfunction.

Procedure

1. On a command line, run the following command to view the URL of RHOSP public endpoints:

   ```
   $ openstack catalog list
   ```

   Record the URL for each HTTPS endpoint that the command returns.

2. For each public endpoint, note the host and the port.

   TIP

   Determine the host of an endpoint by removing the scheme, the port, and the path.

3. For each endpoint, run the following commands to extract the SAN field of the certificate:

   a. Set a **host** variable:

      ```
      $ host=<host_name>
      ```

   b. Set a **port** variable:

      ```
      $ port=<port_number>
      ```

      If the URL of the endpoint does not have a port, use the value **443**.

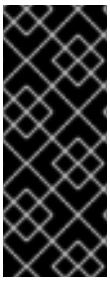
   c. Retrieve the SAN field of the certificate:

```
$ openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
    | openssl x509 -noout -ext subjectAltName
```

Example output

```
X509v3 Subject Alternative Name:
    DNS:your.host.example.net
```

For each endpoint, look for output that resembles the previous example. If there is no output for an endpoint, the certificate of that endpoint is invalid and must be re-issued.



IMPORTANT

You must replace all legacy HTTPS certificates before you install OpenShift Container Platform 4.10 or update a cluster to that version. Legacy certificates are rejected with the following message:

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

CHAPTER 2. PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV OR OVS-DPDK ON OPENSTACK

Before you install a OpenShift Container Platform cluster that uses single-root I/O virtualization (SR-IOV) or Open vSwitch with the Data Plane Development Kit (OVS-DPDK) on Red Hat OpenStack Platform (RHOSP), you must understand the requirements for each technology and then perform preparatory tasks.

2.1. REQUIREMENTS FOR CLUSTERS ON RHOSP THAT USE EITHER SR-IOV OR OVS-DPDK

If you use SR-IOV or OVS-DPDK with your deployment, you must meet the following requirements:

- RHOSP compute nodes must use a flavor that supports huge pages.

2.1.1. Requirements for clusters on RHOSP that use SR-IOV

To use single-root I/O virtualization (SR-IOV) with your deployment, you must meet the following requirements:

- Plan your Red Hat OpenStack Platform (RHOSP) SR-IOV deployment .
- OpenShift Container Platform must support the NICs that you use. For a list of supported NICs, see "About Single Root I/O Virtualization (SR-IOV) hardware networks" in the "Hardware networks" subsection of the "Networking" documentation.
- For each node that will have an attached SR-IOV NIC, your RHOSP cluster must have:
  - One instance from the RHOSP quota
  - One port attached to the machines subnet
  - One port for each SR-IOV Virtual Function
  - A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space
- SR-IOV deployments often employ performance optimizations, such as dedicated or isolated CPUs. For maximum performance, configure your underlying RHOSP deployment to use these optimizations, and then run OpenShift Container Platform compute machines on the optimized infrastructure.
  - For more information about configuring performant RHOSP compute nodes, see Configuring Compute nodes for performance.

2.1.2. Requirements for clusters on RHOSP that use OVS-DPDK

To use Open vSwitch with the Data Plane Development Kit (OVS-DPDK) with your deployment, you must meet the following requirements:

- Plan your Red Hat OpenStack Platform (RHOSP) OVS-DPDK deployment by referring to Planning your OVS-DPDK deployment in the Network Functions Virtualization Planning and Configuration Guide.
- Configure your RHOSP OVS-DPDK deployment according to Configuring an OVS-DPDK deployment in the Network Functions Virtualization Planning and Configuration Guide.

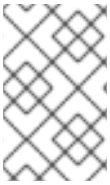
2.2. PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV

You must configure RHOSP before you install a cluster that uses SR-IOV on it.

When installing a cluster using SR-IOV, you must deploy clusters using cgroup v1. For more information, Enabling Linux control group version 1 (cgroup v1) .

2.2.1. Creating SR-IOV networks for compute machines

If your Red Hat OpenStack Platform (RHOSP) deployment supports single root I/O virtualization (SR-IOV), you can provision SR-IOV networks that compute machines run on.



NOTE

The following instructions entail creating an external flat network and an external, VLAN-based network that can be attached to a compute machine. Depending on your RHOSP deployment, other network types might be required.

Prerequisites

- Your cluster supports SR-IOV.

NOTE

If you are unsure about what your cluster supports, review the OpenShift Container Platform SR-IOV hardware networks documentation.

- You created radio and uplink provider networks as part of your RHOSP deployment. The names **radio** and **uplink** are used in all example commands to represent these networks.

Procedure

1. On a command line, create a radio RHOSP network:

```
$ openstack network create radio --provider-physical-network radio --provider-network-type flat --external
```

2. Create an uplink RHOSP network:

```
$ openstack network create uplink --provider-physical-network uplink --provider-network-type vlan --external
```

3. Create a subnet for the radio network:

```
$ openstack subnet create --network radio --subnet-range <radio_network_subnet_range> radio
```

4. Create a subnet for the uplink network:

```
$ openstack subnet create --network uplink --subnet-range <uplink_network_subnet_range> uplink
```

2.3. PREPARING TO INSTALL A CLUSTER THAT USES OVS-DPDK

You must configure RHOSP before you install a cluster that uses SR-IOV on it.

- Complete Creating a flavor and deploying an instance for OVS-DPDK before you install a cluster on RHOSP.

After you perform preinstallation tasks, install your cluster by following the most relevant OpenShift Container Platform on RHOSP installation instructions. Then, perform the tasks under "Next steps" on this page.

2.4. NEXT STEPS

- For either type of deployment:
  - Configure the Node Tuning Operator with huge pages support .
- To complete SR-IOV configuration after you deploy your cluster:
  - Install the SR-IOV Operator.
  - Configure your SR-IOV network device .
  - Create SR-IOV compute machines.
- Consult the following references after you deploy your cluster to improve its performance:
  - A test pod template for clusters that use OVS-DPDK on OpenStack .
  - A test pod template for clusters that use SR-IOV on OpenStack .
  - A performance profile template for clusters that use OVS-DPDK on OpenStack .

CHAPTER 3. INSTALLING A CLUSTER ON OPENSTACK WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.15, you can install a customized cluster on Red Hat OpenStack Platform (RHOSP). To customize the installation, modify parameters in the **install-config.yaml** before you install the cluster.

3.1. PREREQUISITES

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You verified that OpenShift Container Platform 4.15 is compatible with your RHOSP version by using the Supported platforms for OpenShift clusters section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
- You have a storage service installed in RHOSP, such as block storage (Cinder) or object storage (Swift). Object storage is the recommended storage technology for OpenShift Container Platform registry cluster deployment. For more information, see Optimizing storage.
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see Recommended practices for scaling the cluster.
- You have the metadata service enabled in RHOSP.

3.2. RESOURCE GUIDELINES FOR INSTALLING OPENSHIFT CONTAINER PLATFORM ON RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 3.1. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

Resource	Value
Floating IP addresses	3
Ports	15
Routers	1
Subnets	1
RAM	88 GB
vCPUs	22

Resource	Value
Volume storage	275 GB
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

3.2.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

3.2.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

3.2.3. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

3.2.4. Load balancing requirements for user-provisioned infrastructure

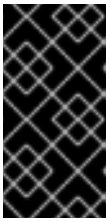
Before you install OpenShift Container Platform, you can provision your own API and application ingress load balancing infrastructure to use in place of the default, internal load balancing solution. In production scenarios, you can deploy the API and application Ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

NOTE

If you want to deploy the API and application Ingress load balancers with a Red Hat Enterprise Linux (RHEL) instance, you must purchase the RHEL subscription separately.

The load balancing infrastructure must meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.



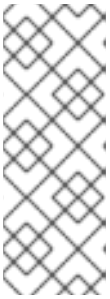
IMPORTANT

Do not configure session persistence for an API load balancer. Configuring session persistence for a Kubernetes API server might cause performance issues from excess application traffic for your OpenShift Container Platform cluster and the Kubernetes API that runs inside the cluster.

Configure the following ports on both the front and back of the load balancers:

Table 3.2. API load balancer

Port	Back-end machines (pool members)	Internal	External	Description
6443	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the **/readyz** endpoint for the API server health check probe.	X	X	Kubernetes API server
22623	Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.	X		Machine config server



NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the **/readyz** endpoint to the removal of the API server instance from the pool. Within the time frame after **/readyz** returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an ingress point for application traffic flowing in from outside the cluster. A working configuration for the Ingress router is required for an OpenShift Container Platform cluster.
   Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP or SSL Passthrough mode.
   - A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

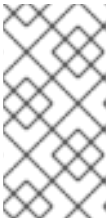
   TIP

   If the true IP address of the client can be seen by the application Ingress load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

   Configure the following ports on both the front and back of the load balancers:

Table 3.3. Application Ingress load balancer

Port	Back-end machines (pool members)	Internal	External	Description
443	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTPS traffic
80	The machines that run the Ingress Controller pods, compute, or worker, by default.	X	X	HTTP traffic



NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

3.2.4.1. Example load balancer configuration for clusters that are deployed with user-managed load balancers

This section provides an example API and application Ingress load balancer configuration that meets the load balancing requirements for clusters that are deployed with user-managed load balancers. The sample is an **/etc/haproxy/haproxy.cfg** configuration for an HAProxy load balancer. The example is not meant to provide advice for choosing one load balancing solution over another.

In the example, the same load balancer is used for the Kubernetes API and application ingress traffic. In production scenarios, you can deploy the API and application ingress load balancers separately so that you can scale the load balancer infrastructure for each in isolation.

NOTE

If you are using HAProxy as a load balancer and SELinux is set to **enforcing**, you must ensure that the HAProxy service can bind to the configured TCP port by running **setsebool -P haproxy_connect_any=1**.

Example 3.1. Sample API and application Ingress load balancer configuration

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
```

```
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1) Port **6443** handles the Kubernetes API traffic and points to the control plane machines.

(2) (4) The bootstrap entries must be in place before the OpenShift Container Platform cluster installation and they must be removed after the bootstrap process is complete.

(3) Port **22623** handles the machine config server traffic and points to the control plane machines.

(5) Port **443** handles the HTTPS traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

(6) Port **80** handles the HTTP traffic and points to the machines that run the Ingress Controller pods. The Ingress Controller pods run on the compute machines by default.

NOTE

If you are deploying a three-node cluster with zero compute nodes, the Ingress Controller pods run on the control plane nodes. In three-node cluster deployments, you must configure your application Ingress load balancer to route HTTP and HTTPS traffic to the control plane nodes.

TIP

If you are using HAProxy as a load balancer, you can check that the **haproxy** process is listening on ports **6443**, **22623**, **443**, and **80** by running **netstat -nltupe** on the HAProxy node.

3.3. INTERNET ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, you require access to the internet to install your cluster.

You must have internet access to:

- 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.

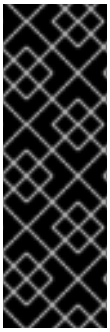
3.4. ENABLING SWIFT ON RHOSP

Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.

IMPORTANT

If the Red Hat OpenStack Platform (RHOSP) object storage service , commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.



IMPORTANT

RHOSP 17 sets the **rgw_max_attr_size** parameter of Ceph RGW to 256 characters. This setting causes issues with uploading container images to the OpenShift Container Platform registry. You must set the value of **rgw_max_attr_size** to at least 1024 characters.

Before installation, check if your RHOSP deployment is affected by this problem. If it is, reconfigure Ceph RGW.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On Ceph RGW, the **account in url** option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

```
$ openstack role add --user <user> --project <project> swiftoperator
```

Your RHOSP deployment can now use Swift for the image registry.

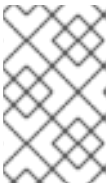
3.5. CONFIGURING AN IMAGE REGISTRY WITH CUSTOM STORAGE ON CLUSTERS THAT RUN ON RHOSP

After you install a cluster on Red Hat OpenStack Platform (RHOSP), you can use a Cinder volume that is in a specific availability zone for registry storage.

Procedure

1. Create a YAML file that specifies the storage class and availability zone to use. For example:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: custom-csi-storageclass
provisioner: cinder.csi.openstack.org
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
parameters:
  availability: <availability_zone_name>
```



NOTE

OpenShift Container Platform does not verify the existence of the availability zone you choose. Verify the name of the availability zone before you apply the configuration.

2. From a command line, apply the configuration:

```
$ oc apply -f <storage_class_file_name>
```

Example output

```
storageclass.storage.k8s.io/custom-csi-storageclass created
```

3. Create a YAML file that specifies a persistent volume claim (PVC) that uses your storage class and the **openshift-image-registry** namespace. For example:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc-imageregistry
  namespace: openshift-image-registry (1)
  annotations:
    imageregistry.openshift.io: "true"
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi (2)
  storageClassName: <your_custom_storage_class> (3)
```

(1) Enter the namespace **openshift-image-registry**. This namespace allows the Cluster Image Registry Operator to consume the PVC.

(2) Optional: Adjust the volume size.

(3) Enter the name of the storage class that you created.

4. From a command line, apply the configuration:

```
$ oc apply -f <pvc_file_name>
```

Example output

```
persistentvolumeclaim/csi-pvc-imageregistry created
```

5. Replace the original persistent volume claim in the image registry configuration with the new claim:

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type 'json' -p='[{"op":
"replace", "path": "/spec/storage/pvc/claim", "value": "csi-pvc-imageregistry"}]'
```

Example output

```
config.imageregistry.operator.openshift.io/cluster patched
```

Over the next several minutes, the configuration is updated.

Verification

To confirm that the registry is using the resources that you defined:

1. Verify that the PVC claim value is identical to the name that you provided in your PVC definition:

```
$ oc get configs.imageregistry.operator.openshift.io/cluster -o yaml
```

Example output

```
...
status:
    ...
    managementState: Managed
    pvc:
      claim: csi-pvc-imageregistry
...
```

2. Verify that the status of the PVC is **Bound**:

```
$ oc get pvc -n openshift-image-registry csi-pvc-imageregistry
```

Example output

```
NAME                    STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS            AGE
csi-pvc-imageregistry   Bound    pvc-72a8f9c9-f462-11e8-b6b6-fa163e18b7b5   100Gi
RWO            custom-csi-storageclass   11m
```

3.6. VERIFYING EXTERNAL NETWORK ACCESS

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries

Procedure

1. Using the RHOSP CLI, verify the name and ID of the 'External' network:

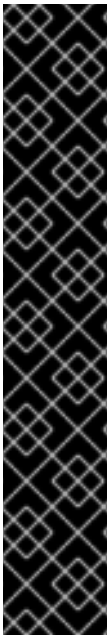
```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
```

```
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network .



IMPORTANT

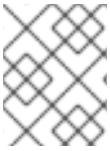
If the external network's CIDR range overlaps one of the default network ranges, you must change the matching network ranges in the **install-config.yaml** file before you start the installation process.

The default network ranges are:

Network	Range
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14

WARNING

If the installation program finds multiple networks with the same name, it sets one of them at random. To avoid this behavior, create unique names for resources in RHOSP.



NOTE

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see Neutron trunk port .

3.7. DEFINING PARAMETERS FOR THE INSTALLATION PROGRAM

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:
   - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.

IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in a separate file from **clouds.yaml**.

- If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see Config files in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:

   a. Copy the certificate authority file to your machine.

   b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

   TIP

   After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:

   a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable

   b. The current directory

   c. A Unix-specific user configuration directory, for example ~/**.config**/**openstack**/**clouds.yaml**

d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**
The installation program searches for **clouds.yaml** in that order.

3.8. SETTING OPENSTACK CLOUD CONTROLLER MANAGER OPTIONS

Optionally, you can edit the OpenStack Cloud Controller Manager (CCM) configuration for your cluster. This configuration controls how OpenShift Container Platform interacts with Red Hat OpenStack Platform (RHOSP).

For a complete list of configuration parameters, see the "OpenStack Cloud Controller Manager reference guide" page in the "Installing on OpenStack" documentation.

Procedure

1. If you have not already generated manifest files for your cluster, generate them by running the following command:

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

2. In a text editor, open the cloud-provider configuration manifest file. For example:

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. Modify the options according to the CCM reference guide.
Configuring Octavia for load balancing is a common case. For example:

```
#...
[LoadBalancer]
lb-provider = "amphora" 1
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 2
create-monitor = True 3
monitor-delay = 10s 4
monitor-timeout = 10s 5
monitor-max-retries = 1 6
#...
```

1 This property sets the Octavia provider that your load balancer uses. It accepts **"ovn"** or **"amphora"** as values. If you choose to use OVN, you must also set **lb-method** to **SOURCE_IP_PORT**.

2 This property is required if you want to use multiple external networks with your cluster. The cloud provider creates floating IP addresses on the network that is specified here.

3 This property controls whether the cloud provider creates health monitors for Octavia load balancers. Set the value to **True** to create health monitors. As of RHOSP 16.2, this feature is only available for the Amphora provider.

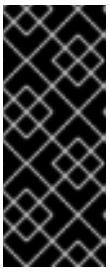
4 This property sets the frequency with which endpoints are monitored. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.

5 This property sets the time that monitoring requests are open before timing out. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.

6 This property defines how many successful monitoring requests are required before a load balancer is marked as online. The value must be an integer. This property is required if the value of the **create-monitor** property is **True**.

IMPORTANT

Prior to saving your changes, verify that the file is structured correctly. Clusters might fail if properties are not placed in the appropriate section.



IMPORTANT

You must set the value of the **create-monitor** property to **True** if you use services that have the value of the **.spec.externalTrafficPolicy** property set to **Local**. The OVN Octavia provider in RHOSP 16.2 does not support health monitors. Therefore, services that have **ETP** parameter values set to **Local** might not respond when the **lb-provider** value is set to **"ovn"**.

4. Save the changes to the file and proceed with installation.

TIP

You can update your cloud provider configuration after you run the installer. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

After you save your changes, your cluster will take some time to reconfigure itself. The process is complete if none of your nodes have a **SchedulingDisabled** status.

3.9. 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 at least 1.2 GB 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.

3.10. CREATING THE INSTALLATION CONFIGURATION FILE

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

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

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> 1
```

-

1 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 **openstack** as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

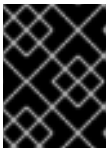
v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

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 OpenStack

3.10.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

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

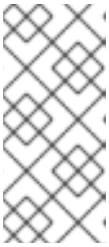
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. 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.
4. If provided, the installation program generates a config map that is named **user-ca-bundle**

5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and

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.

3.10.2. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.

- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.

NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.

IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

3.10.3. Deploying a cluster with bare metal machines

If you want your cluster to use bare metal machines, modify the **install-config.yaml** file. Your cluster can have compute machines running on bare metal.

NOTE

Be sure that your **install-config.yaml** file reflects whether the RHOSP network that you use for bare metal workers supports floating IP addresses or not.

Prerequisites

- The RHOSP Bare Metal service (Ironic) is enabled and accessible via the RHOSP Compute API.
- Bare metal is available as a RHOSP flavor.
- If your cluster runs on an RHOSP version that is more than 16.1.6 and less than 16.2.4, bare metal workers do not function due to a known issue that causes the metadata service to be unavailable for services on OpenShift Container Platform nodes.
- The RHOSP network supports both VM and bare metal server attachment.
- If you want to deploy the machines on a pre-existing network, a RHOSP subnet is provisioned.
- If you want to deploy the machines on an installer-provisioned network, the RHOSP Bare Metal service (Ironic) is able to listen for and interact with Preboot eXecution Environment (PXE) boot machines that run on tenant networks.
- You created an **install-config.yaml** file as part of the OpenShift Container Platform installation process.

Procedure

1. In the **install-config.yaml** file, edit the flavors for machines:

a. Change the value of **compute.platform.openstack.type** to a bare metal flavor.

b. If you want to deploy your machines on a pre-existing network, change the value of **platform.openstack.machinesSubnet** to the RHOSP subnet UUID of the network.

An example bare metal install-config.yaml file

```
compute:
  - architecture: amd64
    hyperthreading: Enabled
    name: worker
    platform:
      openstack:
        type: <bare_metal_compute_flavor> 1
    replicas: 3
...

platform:
    openstack:
      machinesSubnet: <subnet_UUID> 2
...
```

1. Change this value to a bare metal flavor to use for compute machines.
2. If you want to use a pre-existing network, change this value to the UUID of the RHOSP subnet.

Use the updated **install-config.yaml** file to complete the installation process. The compute machines that are created during deployment use the flavor that you added to the file.

NOTE

The installer may time out while waiting for bare metal machines to boot.

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

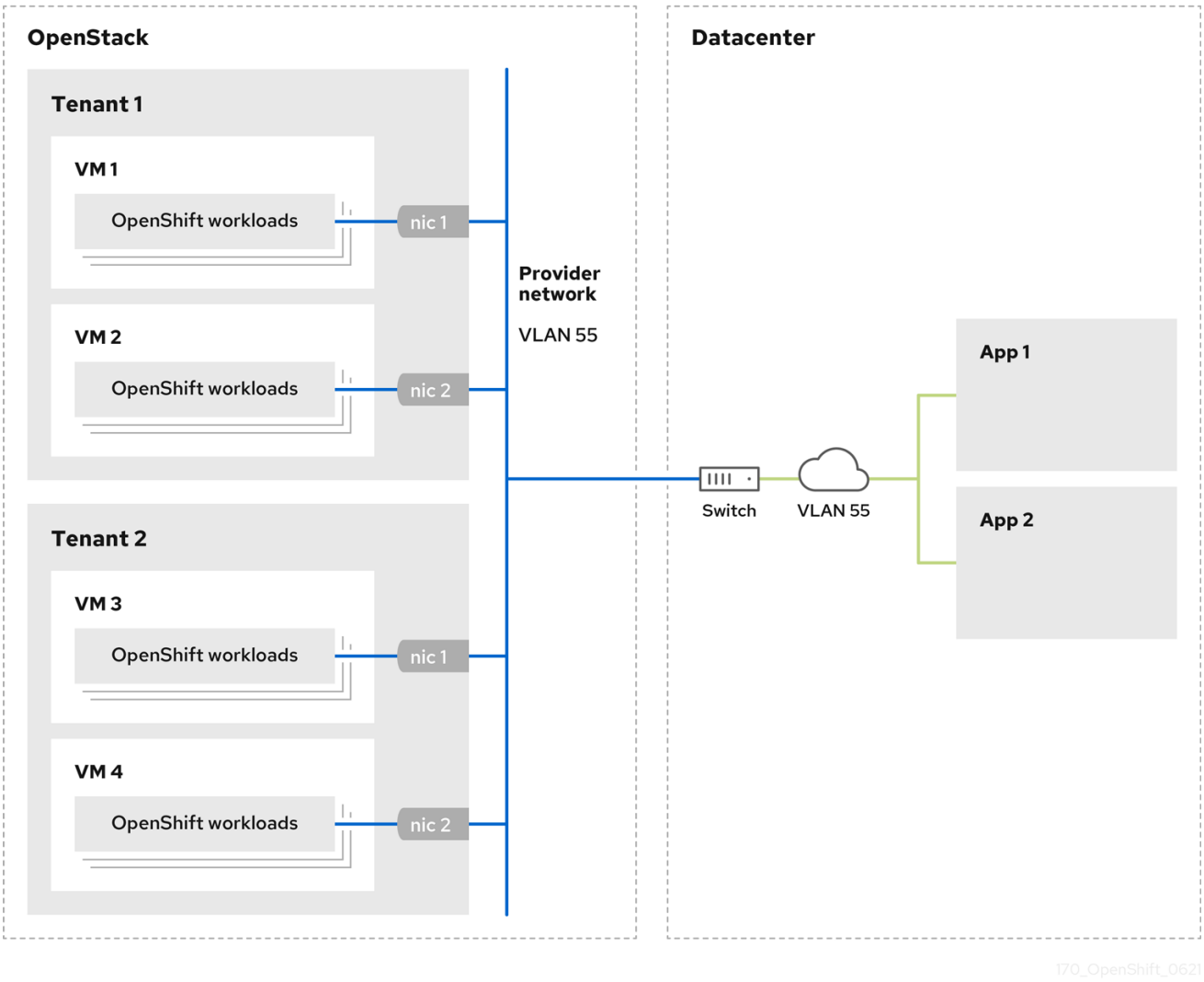
```
$ ./openshift-install wait-for install-complete --log-level debug
```

3.10.4. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

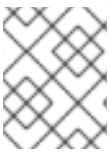
RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).



NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in the RHOSP documentation.

3.10.4.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The RHOSP networking service (Neutron) is enabled and accessible through the RHOSP networking API.
- The RHOSP networking service has the port security and allowed address pairs extensions enabled.

- The provider network can be shared with other tenants.

  TIP

  Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

  TIP

  To create a network for a project that is named "openshift," enter the following command

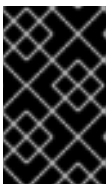
  ```
  $ openstack network create --project openshift
  ```

  To create a subnet for a project that is named "openshift," enter the following command

  ```
  $ openstack subnet create --project openshift
  ```

  To learn more about creating networks on RHOSP, read the provider networks documentation.

  If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.

  IMPORTANT

  Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.
  Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

  ```
  $ openstack subnet create --dhcp --host-route
  destination=169.254.169.254/32,gateway=192.0.2.2 ...
  ```

- Optional: To secure the network, create role-based access control (RBAC) rules that limit network access to a single project.

3.10.4.2. Deploying a cluster that has a primary interface on a provider network

You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.

IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2 In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

TIP

You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see Understanding multiple networks.

3.10.5. Sample customized install-config.yaml file for RHOSP

The following example **install-config.yaml** files demonstrate all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

Example 3.2. Example single stack install-config.yaml file

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

Example 3.3. Example dual stack install-config.yaml file

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.25.0/24
  - cidr: fd2e:6f44:5dd8:c956::/64
  serviceNetwork:
  - 172.30.0.0/16
  - fd02::/112
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiVIPs:
    - 192.168.25.10
    - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
    ingressVIPs:
    - 192.168.25.132
    - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
    controlPlanePort:
      fixedIPs:
      - subnet:
          name: openshift-dual4
      - subnet:
          name: openshift-dual6
      network:
        name: openshift-dual
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

3.10.6. Configuring a cluster with dual-stack networking

You can create a dual-stack cluster on RHOSP. However, the dual-stack configuration is enabled only if you are using an RHOSP network with IPv4 and IPv6 subnets.

NOTE

RHOSP does not support the conversion of an IPv4 single-stack cluster to a dual-stack cluster network.

3.10.6.1. Deploying the dual-stack cluster

For dual-stack networking in OpenShift Container Platform clusters, you can configure IPv4 and IPv6 address endpoints for cluster nodes.

Prerequisites

- You enabled Dynamic Host Configuration Protocol (DHCP) on the subnets.

Procedure

1. Create a network with IPv4 and IPv6 subnets. The available address modes for the **ipv6-ra-mode** and **ipv6-address-mode** fields are: **dhcpv6-stateful**, **dhcpv6-stateless**, and **slaac**.

NOTE

The dual-stack network MTU must accommodate both the minimum MTU for IPv6, which is **1280**, and the OVN-Kubernetes encapsulation overhead, which is **100**.

2. Create the API and Ingress VIPs ports.
3. Add the IPv6 subnet to the router to enable router advertisements. If you are using a provider network, you can enable router advertisements by adding the network as an external gateway, which also enables external connectivity.
4. Choose one of the following **install-config.yaml** configurations:
   a. For an IPv4/IPv6 dual-stack cluster where you set IPv4 as the primary endpoint for your cluster nodes, edit the **install-config.yaml** file in a similar way to the following example:

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
```

```
  name: mycluster
networking:
  machineNetwork: 1
  - cidr: "192.168.25.0/24"
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  clusterNetwork: 2
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  serviceNetwork: 3
  - 172.30.0.0/16
  - fd02::/112
platform:
  openstack:
    ingressVIPs: ['192.168.25.79', 'fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad'] 4
    apiVIPs: ['192.168.25.199', 'fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36'] 5
    controlPlanePort: 6
      fixedIPs: 7
      - subnet: 8
          name: subnet-v4
          id: subnet-v4-id
      - subnet: 9
          name: subnet-v6
          id: subnet-v6-id
      network: 10
        name: dualstack
        id: network-id
```

1 2 3 You must specify an IP address range for both the IPv4 and IPv6 address families.

4 Specify the virtual IP (VIP) address endpoints for the Ingress VIP services to provide an interface to the cluster.

5 Specify the virtual IP (VIP) address endpoints for the API VIP services to provide an interface to the cluster.

6 Specify the dual-stack network details that all of the nodes across the cluster use for their networking needs.

7 The Classless Inter-Domain Routing (CIDR) of any subnet specified in this field must match the CIDRs listed on **networks.machineNetwork**.

8 9 You can specify a value for either **name** or **id**, or both.

10 Specifying the **network** under the **ControlPlanePort** field is optional.

b. For an IPv6/IPv4 dual-stack cluster where you set IPv6 as the primary endpoint for your cluster nodes, edit the **install-config.yaml** file in a similar way to the following example:

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
```

```
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork: 1
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  - cidr: "192.168.25.0/24"
  clusterNetwork: 2
  - cidr: fd01::/48
    hostPrefix: 64
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork: 3
  - fd02::/112
  - 172.30.0.0/16
platform:
  openstack:
    ingressVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad', '192.168.25.79'] 4
    apiVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36', '192.168.25.199'] 5
    controlPlanePort: 6
      fixedIPs: 7
      - subnet: 8
          name: subnet-v6
          id: subnet-v6-id
      - subnet: 9
          name: subnet-v4
          id: subnet-v4-id
      network: 10
        name: dualstack
        id: network-id
```

1 2 3 You must specify an IP address range for both the IPv4 and IPv6 address families.

4 Specify the virtual IP (VIP) address endpoints for the Ingress VIP services to provide an interface to the cluster.

5 Specify the virtual IP (VIP) address endpoints for the API VIP services to provide an interface to the cluster.

6 Specify the dual-stack network details that all the nodes across the cluster use for their networking needs.

7 The CIDR of any subnet specified in this field must match the CIDRs listed on **networks.machineNetwork**.

8 9 You can specify a value for either **name** or **id**, or both.

10 Specifying the **network** under the **controlPlanePort** field is optional.

5. Optional: When you use an installation host in an isolated dual-stack network, the IPv6 address might not be reassigned correctly upon reboot. To resolve this problem on Red Hat Enterprise Linux (RHEL) 8, complete the following steps:

   a. Create a file called **/etc/NetworkManager/system-connections/required-rhel8-ipv6.conf** that includes the following configuration:

   ```
   [connection]
   type=ethernet
   [ipv6]
   addr-gen-mode=eui64
   method=auto
   ```

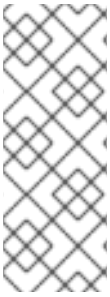
   b. Reboot the installation host.

6. Optional: When you use an installation host in an isolated dual-stack network, the IPv6 address might not be reassigned correctly upon reboot. To resolve this problem on Red Hat Enterprise Linux (RHEL) 9, complete the following steps:

   a. Create a file called **/etc/NetworkManager/conf.d/required-rhel9-ipv6.conf** that includes the following configuration:

   ```
   [connection]
   ipv6.addr-gen-mode=0
   ```

   b. Reboot the installation host.



NOTE

The **ip=dhcp,dhcp6** kernel argument, which is set on all of the nodes, results in a single Network Manager connection profile that is activated on multiple interfaces simultaneously. Because of this behavior, any additional network has the same connection enforced with an identical UUID. If you need an interface-specific configuration, create a new connection profile for that interface so that the default connection is no longer enforced on it.

3.10.7. Installation configuration for a cluster on OpenStack with a user-managed load balancer

The following example **install-config.yaml** file demonstrates how to configure a cluster that uses an external, user-managed load balancer rather than the default internal load balancer.

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
```

```
    platform:
      openstack:
        type: m1.xlarge
    replicas: 3
  metadata:
    name: mycluster
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 192.168.10.0/24
  platform:
    openstack:
      cloud: mycloud
      machinesSubnet: 8586bf1a-cc3c-4d40-bdf6-c243decc603a (1)
      apiVIPs:
      - 192.168.10.5
      ingressVIPs:
      - 192.168.10.7
      loadBalancer:
        type: UserManaged (2)
```

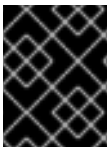
1. Regardless of which load balancer you use, the load balancer is deployed to this subnet.
2. The **UserManaged** value indicates that you are using an user-managed load balancer.

3.11. GENERATING A KEY PAIR FOR CLUSTER NODE SSH ACCESS

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, 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

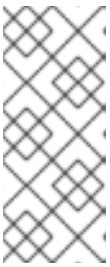
Do not skip this procedure in production environments, where disaster recovery and debugging is required.

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> 1
```

1 Specify 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> 1
```

1 Specify 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.

3.12. ENABLING ACCESS TO THE ENVIRONMENT

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

3.12.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your **/etc/hosts** file:

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

The cluster domain names in the **/etc/hosts** file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the <application_floating_ip>. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

4. Add the FIPs to the **install-config.yaml** file as the values of the following parameters:
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.apiFloatingIP**

If you use these values, you must also enter an external network as the value of the **platform.openstack.externalNetwork** parameter in the **install-config.yaml** file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

3.12.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the **install-config.yaml** file, do not define the following parameters:

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.apiFloatingIP**

If you cannot provide an external network, you can also leave **platform.openstack.externalNetwork**

blank. If you do not provide a value for **platform.openstack.externalNetwork**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.

NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

3.13. 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

- Change to the directory that contains the installation program and initialize the cluster deployment:

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

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

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

Verification

When the cluster deployment completes successfully:

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

IMPORTANT

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

Example output

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

IMPORTANT

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

3.14. VERIFYING CLUSTER STATUS

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

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

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

3. View your cluster's version:

```
$ oc get clusterversion
```

4. View your Operators' status:

```
$ oc get clusteroperator
```

5. View all running pods in the cluster:

```
$ oc get pods -A
```

3.15. LOGGING IN TO THE CLUSTER BY USING THE CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

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

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

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

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.16. TELEMETRY ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered 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.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

3.17. NEXT STEPS

- Customize your cluster
- Remote health reporting.
- configure ingress cluster traffic by using a node port

CHAPTER 4. INSTALLING A CLUSTER ON OPENSTACK ON YOUR OWN INFRASTRUCTURE

In OpenShift Container Platform version 4.15, you can install a cluster on Red Hat OpenStack Platform (RHOSP) that runs on user-provisioned infrastructure.

Using your own infrastructure allows you to integrate your cluster with existing infrastructure and modifications. The process requires more labor on your part than installer-provisioned installations, because you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. However, Red Hat provides Ansible playbooks to help you in the deployment process.

4.1. PREREQUISITES

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You verified that OpenShift Container Platform 4.15 is compatible with your RHOSP version by using the Supported platforms for OpenShift clusters section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
- You have an RHOSP account where you want to install OpenShift Container Platform.
- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see Recommended practices for scaling the cluster .
- On the machine from which you run the installation program, you have:
  - A single directory in which you can keep the files you create during the installation process
  - Python 3

4.2. INTERNET ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, you require access to the internet to install your cluster.

You must have internet access to:

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

IMPORTANT

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

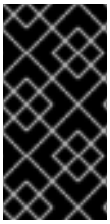
4.3. RESOURCE GUIDELINES FOR INSTALLING OPENSHIFT CONTAINER PLATFORM ON RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 4.1. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

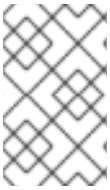
Resource	Value
Floating IP addresses	3
Ports	15
Routers	1
Subnets	1
RAM	88 GB
vCPUs	22
Volume storage	275 GB
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.



IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.



NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

4.3.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

4.3.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

4.3.3. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

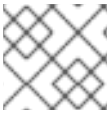
- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

4.4. DOWNLOADING PLAYBOOK DEPENDENCIES

The Ansible playbooks that simplify the installation process on user-provisioned infrastructure require several ansible collections and Python modules. On the machine where you will run the installation program, add the Red Hat OpenStack Platform (RHOSP) repositories and then install the packages.

The following dependencies are required:

- Python modules:
  - **openstackclient**
  - **openstacksdk**
  - **netaddr**
  - **pip**
- Ansible collections:
  - **ansible-collections-openstack**, which installs Ansible Core
  - **ansible-collection-community-general**
  - **ansible-collection-ansible-netcommon**



NOTE

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:
   a. Register with Red Hat Subscription Manager:

```
$ sudo subscription-manager register # If not done already
```

   b. Pull the latest subscription data:

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. Disable the current repositories:

```
$ sudo subscription-manager repos --disable=* # If not done already
```

d. Add the required repositories:

```
$ sudo subscription-manager repos \
  --enable=rhel-9-for-x86_64-appstream-rpms \
  --enable=rhel-9-for-x86_64-baseos-rpms \
  --enable=openstack-17.1-for-rhel-9-x86_64-rpms
```

2. Install the modules:

```
$ sudo dnf install ansible-collection-ansible-netcommon \
    ansible-collection-community-general \
    ansible-collections-openstack \
    python3-netaddr \
    python3-openstackclient \
    python3-openstacksdk \
    python3-pip
```

3. Ensure that the **python** command points to **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

4.5. DOWNLOADING THE INSTALLATION PLAYBOOKS

Download Ansible playbooks that you can use to install OpenShift Container Platform on your own Red Hat OpenStack Platform (RHOSP) infrastructure.

Prerequisites

- The curl command-line tool is available on your machine.

Procedure

- To download the playbooks to your working directory, run the following script from a command line:

```
$ xargs -n 1 curl -O <<< '
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/bootstrap.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/common.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/compute-nodes.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/control-
plane.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
bootstrap.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
compute-nodes.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
control-plane.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
```

```
network.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
security-groups.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/down-
containers.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/inventory.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/network.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.15/upi/openstack/security-groups.yaml
      https://raw.githubusercontent.com/openshift/installer/release-4.15/upi/openstack/update-
network-resources.yaml'
```

The playbooks are downloaded to your machine.

IMPORTANT

During the installation process, you can modify the playbooks to configure your deployment.

Retain all playbooks for the life of your cluster. You must have the playbooks to remove your OpenShift Container Platform cluster from RHOSP.

IMPORTANT

You must match any edits you make in the **bootstrap.yaml**, **compute-nodes.yaml**, **control-plane.yaml**, **network.yaml**, and **security-groups.yaml** files to the corresponding playbooks that are prefixed with **down-**. For example, edits to the **bootstrap.yaml** file must be reflected in the **down-bootstrap.yaml** file, too. If you do not edit both files, the supported cluster removal process will fail.

4.6. 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 at least 1.2 GB of local disk space.

Procedure

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

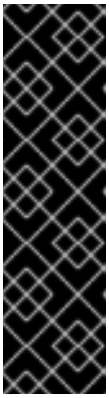
   TIP

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

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

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

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



IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.

- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

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

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

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

TIP

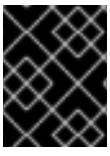
Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

4.7. GENERATING A KEY PAIR FOR CLUSTER NODE SSH ACCESS

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

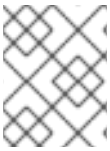
After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

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



IMPORTANT

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



NOTE

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

Procedure

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

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 Specify 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

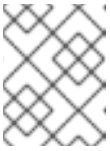
On some distributions, default SSH private key identities such as ~/**.ssh/id_rsa** and ~/**.ssh/id_dsa** are managed automatically.

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> (1)
```

(1) Specify 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.

4.8. CREATING THE RED HAT ENTERPRISE LINUX COREOS (RHCOS) IMAGE

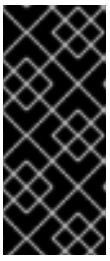
The OpenShift Container Platform installation program requires that a Red Hat Enterprise Linux CoreOS (RHCOS) image be present in the Red Hat OpenStack Platform (RHOSP) cluster. Retrieve the latest RHCOS image, then upload it using the RHOSP CLI.

Prerequisites

- The RHOSP CLI is installed.

Procedure

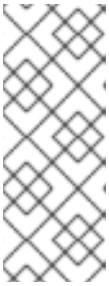
1. Log in to the Red Hat Customer Portal's Product Downloads page.
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.15 for Red Hat Enterprise Linux (RHEL) 8.



IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the *Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)* .
4. Decompress the image.



NOTE

You must decompress the RHOSP image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like **.gz** or **.tgz**. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. From the image that you downloaded, create an image that is named **rhcos** in your cluster by using the RHOSP CLI:

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-${RHCOS_VERSION}-openstack.qcow2 rhcos
```

IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either **.raw** or **.qcow2** formats. If you use Ceph, you must use the **.raw** format.

WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

After you upload the image to RHOSP, it is usable in the installation process.

4.9. VERIFYING EXTERNAL NETWORK ACCESS

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack's networking service to have DHCP agents forward instances' DNS queries

Procedure

1. Using the RHOSP CLI, verify the name and ID of the 'External' network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+--------------------------------------+----------------+-------------+
```

```
| ID                                   | Name          | Router Type |
+--------------------------------------+---------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+---------------+-------------+
```

A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network .

NOTE

If the Neutron trunk service plugin is enabled, a trunk port is created by default. For more information, see Neutron trunk port .

4.10. ENABLING ACCESS TO THE ENVIRONMENT

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

4.10.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API, cluster applications, and the bootstrap process.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. By using the Red Hat OpenStack Platform (RHOSP) CLI, create the bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your **/etc/hosts** file:

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

The cluster domain names in the **/etc/hosts** file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the <application_floating_ip>. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

5. Add the FIPs to the **inventory.yaml** file as the values of the following variables:
   - **os_api_fip**
   - **os_bootstrap_fip**
   - **os_ingress_fip**

If you use these values, you must also enter an external network as the value of the **os_external_network** variable in the **inventory.yaml** file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

4.10.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

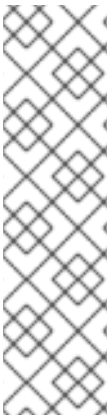
In the **inventory.yaml** file, do not define the following variables:

- **os_api_fip**
- **os_bootstrap_fip**

- **os_ingress_fip**

If you cannot provide an external network, you can also leave **os_external_network** blank. If you do not provide a value for **os_external_network**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. Later in the installation process, when you create network resources, you must configure external connectivity on your own.

If you run the installer with the **wait-for** command from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

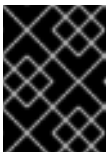
4.11. DEFINING PARAMETERS FOR THE INSTALLATION PROGRAM

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:

   - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.

     IMPORTANT

     Remember to add a password to the **auth** field. You can also keep secrets in a separate file from **clouds.yaml**.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see Config files in the RHOSP documentation.

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: <username>
           password: <password>
           user_domain_name: Default
     ```

```
    project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:

   a. Copy the certificate authority file to your machine.

   b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

   TIP

   After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:

   a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable

   b. The current directory

   c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**

   d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**
   The installation program searches for **clouds.yaml** in that order.

4.12. CREATING NETWORK RESOURCES ON RHOSP

Create the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires. To save time, run supplied Ansible playbooks that generate security groups, networks, subnets, routers, and ports.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".

Procedure

1. For a dual stack cluster deployment, edit the **inventory.yaml** file and uncomment the **os_subnet6** attribute.

2. To ensure that your network resources have unique names on the RHOSP deployment, create an environment variable and JSON file for use in the Ansible playbooks:

   a. Create an environment variable that has a unique name value by running the following command:

   ```
   $ export OS_NET_ID="openshift-$(dd if=/dev/urandom count=4 bs=1 2>/dev/null
   |hexdump -e '"%02x"')"
   ```

   b. Verify that the variable is set by running the following command on a command line:

   ```
   $ echo $OS_NET_ID
   ```

   c. Create a JSON object that includes the variable in a file called **netid.json** by running the following command:

   ```
   $ echo "{\"os_net_id\": \"$OS_NET_ID\"}" | tee netid.json
   ```

3. On a command line, create the network resources by running the following command:

   ```
   $ ansible-playbook -i inventory.yaml network.yaml
   ```

NOTE

The API and Ingress VIP fields will be overwritten in the **inventory.yaml** playbook with the IP addresses assigned to the network ports.

NOTE

The resources created by the **network.yaml** playbook are deleted by the **down-network.yaml** playbook.

4.13. CREATING THE INSTALLATION CONFIGURATION FILE

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

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

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> 1
```

1 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 **openstack** as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

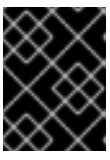
v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

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.

You now have the file **install-config.yaml** in the directory that you specified.

Additional resources

- Installation configuration parameters for OpenStack

4.13.1. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet's GUID is passed as the value of **platform.openstack.machinesSubnet** in the **install-config.yaml** file.

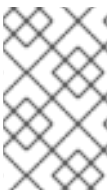
This subnet is used as the cluster's primary subnet. By default, nodes and ports are created on it. You can create nodes and ports on a different RHOSP subnet by setting the value of the **platform.openstack.machinesSubnet** property to the subnet's UUID.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that your configuration meets the following requirements:

- The subnet that is used by **platform.openstack.machinesSubnet** has DHCP enabled.
- The CIDR of **platform.openstack.machinesSubnet** matches the CIDR of **networking.machineNetwork**.
- The installation program user has permission to create ports on this network, including ports with fixed IP addresses.

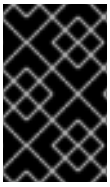
Clusters that use custom subnets have the following limitations:

- If you plan to install a cluster that uses floating IP addresses, the **platform.openstack.machinesSubnet** subnet must be attached to a router that is connected to the **externalNetwork** network.
- If the **platform.openstack.machinesSubnet** value is set in the **install-config.yaml** file, the installation program does not create a private network or subnet for your RHOSP machines.
- You cannot use the **platform.openstack.externalDNS** property at the same time as a custom subnet. To add DNS to a cluster that uses a custom subnet, configure DNS on the RHOSP network.



NOTE

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network's CIDR block. To override these default values, set values for **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** that are outside of the DHCP allocation pool.



IMPORTANT

The CIDR ranges for networks are not adjustable after cluster installation. Red Hat does not provide direct guidance on determining the range during cluster installation because it requires careful consideration of the number of created pods per namespace.

4.13.2. Sample customized install-config.yaml file for RHOSP

The following example **install-config.yaml** files demonstrate all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

Example 4.1. Example single stack install-config.yaml file

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

Example 4.2. Example dual stack install-config.yaml file

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
```

```
    replicas: 3
  metadata:
    name: example
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    - cidr: fd01::/48
      hostPrefix: 64
    machineNetwork:
    - cidr: 192.168.25.0/24
    - cidr: fd2e:6f44:5dd8:c956::/64
    serviceNetwork:
    - 172.30.0.0/16
    - fd02::/112
    networkType: OVNKubernetes
  platform:
    openstack:
      cloud: mycloud
      externalNetwork: external
      computeFlavor: m1.xlarge
      apiVIPs:
      - 192.168.25.10
      - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
      ingressVIPs:
      - 192.168.25.132
      - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
      controlPlanePort:
        fixedIPs:
        - subnet:
            name: openshift-dual4
        - subnet:
            name: openshift-dual6
        network:
          name: openshift-dual
  fips: false
  pullSecret: '{"auths": ...}'
  sshKey: ssh-ed25519 AAAA...
```

4.13.3. Setting a custom subnet for machines

The IP range that the installation program uses by default might not match the Neutron subnet that you create when you install OpenShift Container Platform. If necessary, update the CIDR value for new machines by editing the installation configuration file.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.
- You have Python 3 installed.

Procedure

1. On a command line, browse to the directory that contains the **install-config.yaml** and **inventory.yaml** files.

2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:

   - To set the value by using a script, run the following command:

```
$ python -c 'import os
import sys
import yaml
import re
re_os_net_id = re.compile(r"{{\s*os_net_id\s*}}")
os_net_id = os.getenv("OS_NET_ID")
path = "common.yaml"
facts = None
for _dict in yaml.safe_load(open(path))[0]["tasks"]:
    if "os_network" in _dict.get("set_fact", {}):
        facts = _dict["set_fact"]
        break
if not facts:
    print("Cannot find `os_network` in common.yaml file. Make sure OpenStack resource
names are defined in one of the tasks.")
    sys.exit(1)
os_network = re_os_net_id.sub(os_net_id, facts["os_network"])
os_subnet = re_os_net_id.sub(os_net_id, facts["os_subnet"])
path = "install-config.yaml"
data = yaml.safe_load(open(path))
inventory = yaml.safe_load(open("inventory.yaml"))["all"]["hosts"]["localhost"]
machine_net = [{"cidr": inventory["os_subnet_range"]}]
api_vips = [inventory["os_apiVIP"]]
ingress_vips = [inventory["os_ingressVIP"]]
ctrl_plane_port = {"network": {"name": os_network}, "fixedIPs": [{"subnet": {"name":
os_subnet}}]}
if inventory.get("os_subnet6_range"): (1)
    os_subnet6 = re_os_net_id.sub(os_net_id, facts["os_subnet6"])
    machine_net.append({"cidr": inventory["os_subnet6_range"]})
    api_vips.append(inventory["os_apiVIP6"])
    ingress_vips.append(inventory["os_ingressVIP6"])
    data["networking"]["networkType"] = "OVNKubernetes"
    data["networking"]["clusterNetwork"].append({"cidr": inventory["cluster_network6_cidr"],
"hostPrefix": inventory["cluster_network6_prefix"]})
    data["networking"]["serviceNetwork"].append(inventory["service_subnet6_range"])
    ctrl_plane_port["fixedIPs"].append({"subnet": {"name": os_subnet6}})
data["networking"]["machineNetwork"] = machine_net
data["platform"]["openstack"]["apiVIPs"] = api_vips
data["platform"]["openstack"]["ingressVIPs"] = ingress_vips
data["platform"]["openstack"]["controlPlanePort"] = ctrl_plane_port
del data["platform"]["openstack"]["externalDNS"]
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

(1) Applies to dual stack (IPv4/IPv6) environments.

4.13.4. Emptying compute machine pools

To proceed with an installation that uses your own infrastructure, set the number of compute machines in the installation configuration file to zero. Later, you create these machines manually.

Prerequisites

- You have the **install-config.yaml** file that was generated by the OpenShift Container Platform installation program.

Procedure

1. On a command line, browse to the directory that contains **install-config.yaml**.
2. From that directory, either run a script to edit the **install-config.yaml** file or update the file manually:
   - To set the value by using a script, run:

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

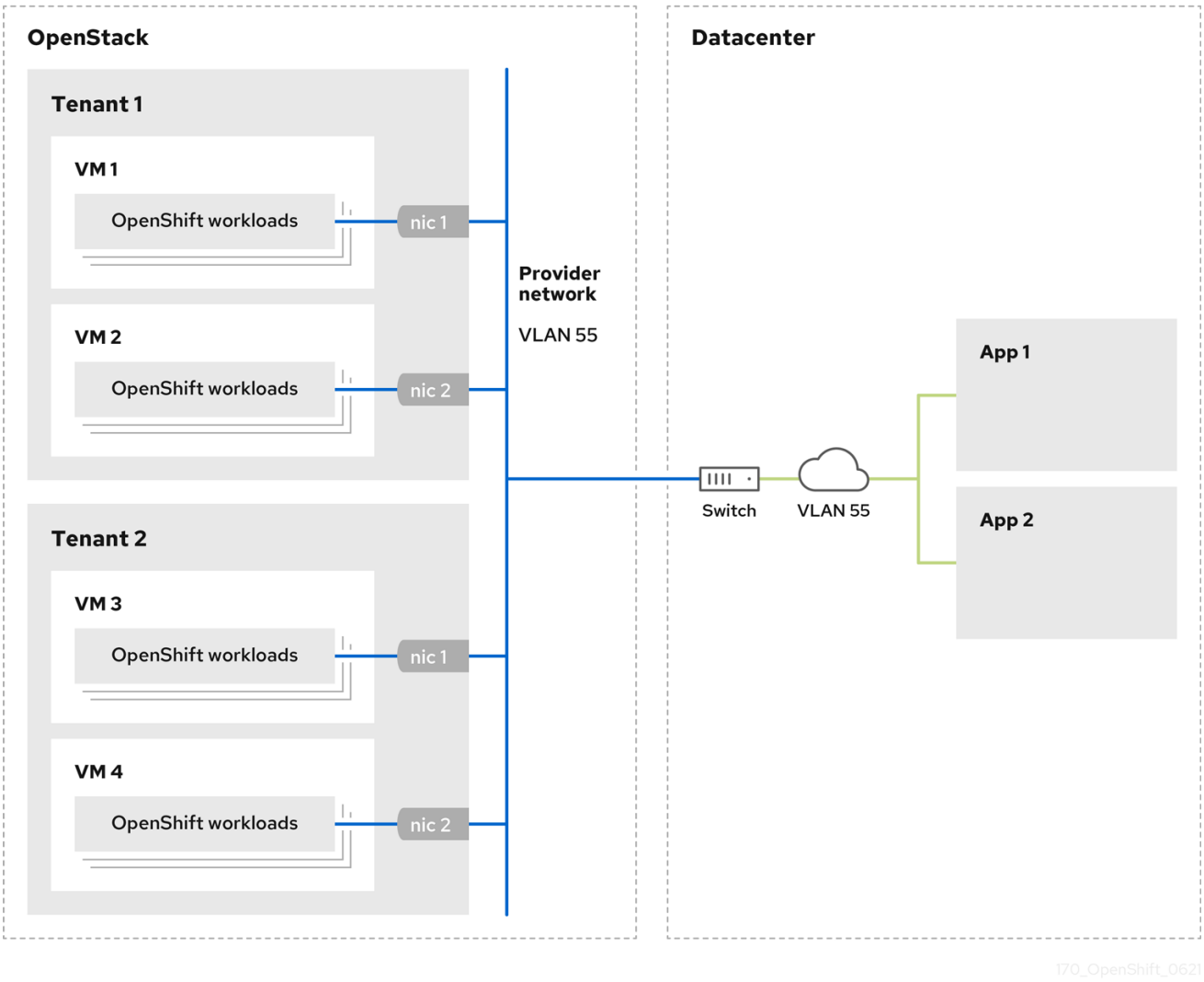
   - To set the value manually, open the file and set the value of **compute.<first entry>.replicas** to **0**.

4.13.5. Cluster deployment on RHOSP provider networks

You can deploy your OpenShift Container Platform clusters on Red Hat OpenStack Platform (RHOSP) with a primary network interface on a provider network. Provider networks are commonly used to give projects direct access to a public network that can be used to reach the internet. You can also share provider networks among projects as part of the network creation process.

RHOSP provider networks map directly to an existing physical network in the data center. A RHOSP administrator must create them.

In the following example, OpenShift Container Platform workloads are connected to a data center by using a provider network:



OpenShift Container Platform clusters that are installed on provider networks do not require tenant networks or floating IP addresses. The installer does not create these resources during installation.

Example provider network types include flat (untagged) and VLAN (802.1Q tagged).

NOTE

A cluster can support as many provider network connections as the network type allows. For example, VLAN networks typically support up to 4096 connections.

You can learn more about provider and tenant networks in the RHOSP documentation.

4.13.5.1. RHOSP provider network requirements for cluster installation

Before you install an OpenShift Container Platform cluster, your Red Hat OpenStack Platform (RHOSP) deployment and provider network must meet a number of conditions:

- The RHOSP networking service (Neutron) is enabled and accessible through the RHOSP networking API.
- The RHOSP networking service has the port security and allowed address pairs extensions enabled.

- The provider network can be shared with other tenants.

  TIP

  Use the **openstack network create** command with the **--share** flag to create a network that can be shared.

- The RHOSP project that you use to install the cluster must own the provider network, as well as an appropriate subnet.

  TIP

  To create a network for a project that is named "openshift," enter the following command

  ```
  $ openstack network create --project openshift
  ```

  To create a subnet for a project that is named "openshift," enter the following command

  ```
  $ openstack subnet create --project openshift
  ```

  To learn more about creating networks on RHOSP, read the provider networks documentation.

  If the cluster is owned by the **admin** user, you must run the installer as that user to create ports on the network.

  IMPORTANT

  Provider networks must be owned by the RHOSP project that is used to create the cluster. If they are not, the RHOSP Compute service (Nova) cannot request a port from that network.

- Verify that the provider network can reach the RHOSP metadata service IP address, which is **169.254.169.254** by default.
  Depending on your RHOSP SDN and networking service configuration, you might need to provide the route when you create the subnet. For example:

  ```
  $ openstack subnet create --dhcp --host-route
  destination=169.254.169.254/32,gateway=192.0.2.2 ...
  ```

- Optional: To secure the network, create role-based access control (RBAC) rules that limit network access to a single project.

4.13.5.2. Deploying a cluster that has a primary interface on a provider network

You can deploy an OpenShift Container Platform cluster that has its primary network interface on an Red Hat OpenStack Platform (RHOSP) provider network.

Prerequisites

- Your Red Hat OpenStack Platform (RHOSP) deployment is configured as described by "RHOSP provider network requirements for cluster installation".

Procedure

1. In a text editor, open the **install-config.yaml** file.
2. Set the value of the **platform.openstack.apiVIPs** property to the IP address for the API VIP.
3. Set the value of the **platform.openstack.ingressVIPs** property to the IP address for the Ingress VIP.
4. Set the value of the **platform.openstack.machinesSubnet** property to the UUID of the provider network subnet.
5. Set the value of the **networking.machineNetwork.cidr** property to the CIDR block of the provider network subnet.

IMPORTANT

The **platform.openstack.apiVIPs** and **platform.openstack.ingressVIPs** properties must both be unassigned IP addresses from the **networking.machineNetwork.cidr** block.

Section of an installation configuration file for a cluster that relies on a RHOSP provider network

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2 In OpenShift Container Platform 4.12 and later, the **apiVIP** and **ingressVIP** configuration settings are deprecated. Instead, use a list format to enter values in the **apiVIPs** and **ingressVIPs** configuration settings.

WARNING

You cannot set the **platform.openstack.externalNetwork** or **platform.openstack.externalDNS** parameters while using a provider network for the primary network interface.

When you deploy the cluster, the installer uses the **install-config.yaml** file to deploy the cluster on the provider network.

TIP

You can add additional networks, including provider networks, to the **platform.openstack.additionalNetworkIDs** list.

After you deploy your cluster, you can attach pods to additional networks. For more information, see Understanding multiple networks.

4.14. CREATING THE KUBERNETES MANIFEST AND IGNITION CONFIG FILES

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

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

IMPORTANT

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

Prerequisites

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

Procedure

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

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

1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines, compute machine sets, and control plane machine sets:

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

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

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

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

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

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

   c. Save and exit the file.

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

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

1 For **<installation_directory>**, specify the same installation directory.

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

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

5. Export the metadata file's **infraID** key as an environment variable:

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

TIP

Extract the **infraID** key from **metadata.json** and use it as a prefix for all of the RHOSP resources that you create. By doing so, you avoid name conflicts when making multiple deployments in the same project.

4.15. PREPARING THE BOOTSTRAP IGNITION FILES

The OpenShift Container Platform installation process relies on bootstrap machines that are created from a bootstrap Ignition configuration file.

Edit the file and upload it. Then, create a secondary bootstrap Ignition configuration file that Red Hat OpenStack Platform (RHOSP) uses to download the primary file.

Prerequisites

- You have the bootstrap Ignition file that the installer program generates, **bootstrap.ign**.
- The infrastructure ID from the installer's metadata file is set as an environment variable (**$INFRA_ID**).
  - If the variable is not set, see **Creating the Kubernetes manifest and Ignition config files**
- You have an HTTP(S)-accessible way to store the bootstrap Ignition file.
  - The documented procedure uses the RHOSP image service (Glance), but you can also use the RHOSP storage service (Swift), Amazon S3, an internal HTTP server, or an ad hoc Nova server.

Procedure

1. Run the following Python script. The script modifies the bootstrap Ignition file to set the hostname and, if available, CA certificate file when it runs:

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

    files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
```

```
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. Using the RHOSP CLI, create an image that uses the bootstrap Ignition file:

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>
```

3. Get the image's details:

```
$ openstack image show <image_name>
```

Make a note of the **file** value; it follows the pattern **v2/images/<image_ID>/file**.

NOTE

Verify that the image you created is active.

4. Retrieve the image service's public address:

```
$ openstack catalog show image
```

5. Combine the public address with the image **file** value and save the result as the storage location. The location follows the pattern **<image_service_public_URL>/v2/images/<image_ID>/file**.

6. Generate an auth token and save the token ID:

```
$ openstack token issue -c id -f value
```

7. Insert the following content into a file called **$INFRA_ID-bootstrap-ignition.json** and edit the placeholders to match your own values:

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
```

```
          "certificateAuthorities": [{
            "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" (4)
          }]
        }
      },
      "version": "3.2.0"
    }
}
```

(1) Replace the value of **ignition.config.merge.source** with the bootstrap Ignition file storage URL.

(2) Set **name** in **httpHeaders** to **"X-Auth-Token"**.

(3) Set **value** in **httpHeaders** to your token's ID.

(4) If the bootstrap Ignition file server uses a self-signed certificate, include the base64-encoded certificate.

8. Save the secondary Ignition config file.

The bootstrap Ignition data will be passed to RHOSP during installation.

WARNING

The bootstrap Ignition file contains sensitive information, like **clouds.yaml** credentials. Ensure that you store it in a secure place, and delete it after you complete the installation process.

4.16. CREATING CONTROL PLANE IGNITION CONFIG FILES ON RHOSP

Installing OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) on your own infrastructure requires control plane Ignition config files. You must create multiple config files.

NOTE

As with the bootstrap Ignition configuration, you must explicitly define a hostname for each control plane machine.

Prerequisites

- The infrastructure ID from the installation program's metadata file is set as an environment variable (**$INFRA_ID**).
  - If the variable is not set, see "Creating the Kubernetes manifest and Ignition config files".

Procedure

- On a command line, run the following Python script:

```
$ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
    python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});
files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
json.dump(ignition, sys.stdout)" <master.ign >"$INFRA_ID-master-$index-ignition.json"
done
```

You now have three control plane Ignition files: **<INFRA_ID>-master-0-ignition.json**, **<INFRA_ID>-master-1-ignition.json**, and **<INFRA_ID>-master-2-ignition.json**.

4.17. UPDATING NETWORK RESOURCES ON RHOSP

Update the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".

Procedure

1. Optional: Add an external network value to the **inventory.yaml** playbook:

Example external network value in the inventory.yaml Ansible Playbook

```
...
    # The public network providing connectivity to the cluster. If not
    # provided, the cluster external connectivity must be provided in another
    # way.

    # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
    os_external_network: 'external'
...
```

IMPORTANT

If you did not provide a value for **os_external_network** in the **inventory.yaml** file, you must ensure that VMs can access Glance and an external connection yourself.

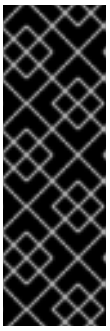
2. Optional: Add external network and floating IP (FIP) address values to the **inventory.yaml** playbook:

Example FIP values in the inventory.yaml Ansible Playbook

```
...
    # OpenShift API floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the Control Plane to
    # serve the OpenShift API.
    os_api_fip: '203.0.113.23'

    # OpenShift Ingress floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the worker nodes to serve
    # the applications.
    os_ingress_fip: '203.0.113.19'

    # If this value is non-empty, the corresponding floating IP will be
    # attached to the bootstrap machine. This is needed for collecting logs
    # in case of install failure.
    os_bootstrap_fip: '203.0.113.20'
```



IMPORTANT

If you do not define values for **os_api_fip** and **os_ingress_fip**, you must perform postinstallation network configuration.

If you do not define a value for **os_bootstrap_fip**, the installation program cannot download debugging information from failed installations.

See "Enabling access to the environment" for more information.

3. On a command line, create security groups by running the **security-groups.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. On a command line, update the network resources by running the **update-network-resources.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml update-network-resources.yaml 1
```

1 This playbook will add tags to the network, subnets, ports, and router. It also attaches floating IP addresses to the API and Ingress ports and sets the security groups for those ports.

5. Optional: If you want to control the default resolvers that Nova servers use, run the RHOSP CLI command:

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

6. Optional: You can use the **inventory.yaml** file that you created to customize your installation. For example, you can deploy a cluster that uses bare metal machines.

4.17.1. Deploying a cluster with bare metal machines

If you want your cluster to use bare metal machines, modify the **inventory.yaml** file. Your cluster can have compute machines running on bare metal.

NOTE

Be sure that your **install-config.yaml** file reflects whether the RHOSP network that you use for bare metal workers supports floating IP addresses or not.

Prerequisites

- The RHOSP Bare Metal service (Ironic) is enabled and accessible via the RHOSP Compute API.
- Bare metal is available as a RHOSP flavor.
- If your cluster runs on an RHOSP version that is more than 16.1.6 and less than 16.2.4, bare metal workers do not function due to a known issue that causes the metadata service to be unavailable for services on OpenShift Container Platform nodes.
- The RHOSP network supports both VM and bare metal server attachment.
- If you want to deploy the machines on a pre-existing network, a RHOSP subnet is provisioned.
- If you want to deploy the machines on an installer-provisioned network, the RHOSP Bare Metal service (Ironic) is able to listen for and interact with Preboot eXecution Environment (PXE) boot machines that run on tenant networks.
- You created an **inventory.yaml** file as part of the OpenShift Container Platform installation process.

Procedure

1. In the **inventory.yaml** file, edit the flavors for machines:
   a. Change the value of **os_flavor_worker** to a bare metal flavor.

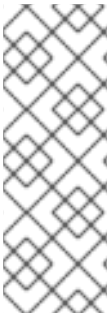
An example bare metal inventory.yaml file

```
all:
  hosts:
    localhost:
      ansible_connection: local
      ansible_python_interpreter: "{{ansible_playbook_python}}"

      # User-provided values
      os_subnet_range: '10.0.0.0/16'
      os_flavor_master: 'my-vm-flavor'
      os_flavor_worker: 'my-bare-metal-flavor' (1)
      os_image_rhcos: 'rhcos'
      os_external_network: 'external'
...
```

(1) Change this value to a bare metal flavor to use for compute machines.

Use the updated **inventory.yaml** file to complete the installation process. Machines that are created during deployment use the flavor that you added to the file.



NOTE

The installer may time out while waiting for bare metal machines to boot.

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
```

4.18. CREATING THE BOOTSTRAP MACHINE ON RHOSP

Create a bootstrap machine and give it the network access it needs to run on Red Hat OpenStack Platform (RHOSP). Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **bootstrap.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **bootstrap.yaml** playbook:

   ```
   $ ansible-playbook -i inventory.yaml bootstrap.yaml
   ```

3. After the bootstrap server is active, view the logs to verify that the Ignition files were received:

   ```
   $ openstack console log show "$INFRA_ID-bootstrap"
   ```

4.19. CREATING THE CONTROL PLANE MACHINES ON RHOSP

Create three control plane machines by using the Ignition config files that you generated. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".

- The infrastructure ID from the installation program's metadata file is set as an environment variable (**$INFRA_ID**).
- The **inventory.yaml**, **common.yaml**, and **control-plane.yaml** Ansible playbooks are in a common directory.
- You have the three Ignition files that were created in "Creating control plane Ignition config files".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. If the control plane Ignition config files aren't already in your working directory, copy them into it.
3. On a command line, run the **control-plane.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. Run the following command to monitor the bootstrapping process:

```
$ openshift-install wait-for bootstrap-complete
```

You will see messages that confirm that the control plane machines are running and have joined the cluster:

```
INFO API v1.28.5 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

4.20. LOGGING IN TO THE CLUSTER BY USING THE CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

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

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

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

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

4.21. DELETING BOOTSTRAP RESOURCES FROM RHOSP

Delete the bootstrap resources that you no longer need.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".
- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **down-bootstrap.yaml** Ansible playbooks are in a common directory.
- The control plane machines are running.
  - If you do not know the status of the machines, see "Verifying cluster status".

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the **down-bootstrap.yaml** playbook:

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

The bootstrap port, server, and floating IP address are deleted.

WARNING

If you did not disable the bootstrap Ignition file URL earlier, do so now.

4.22. CREATING COMPUTE MACHINES ON RHOSP

After standing up the control plane, create compute machines. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies".

- You downloaded the playbooks in "Downloading the installation playbooks".
- The **inventory.yaml**, **common.yaml**, and **compute-nodes.yaml** Ansible playbooks are in a common directory.
- The **metadata.json** file that the installation program created is in the same directory as the Ansible playbooks.
- The control plane is active.

Procedure

1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

Next steps

- Approve the certificate signing requests for the machines.

4.23. APPROVING THE CERTIFICATE SIGNING REQUESTS FOR YOUR MACHINES

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.28.5
master-1 Ready    master  63m  v1.28.5
master-2 Ready    master  64m  v1.28.5
```

The output lists all of the machines that you created.

NOTE

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

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

```
$ oc get csr
```

Example output

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

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

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

NOTE

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

NOTE

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

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

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

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

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

NOTE

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

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

```
$ oc get csr
```

Example output

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

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

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

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

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

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

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

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.28.5
master-1  Ready     master  73m  v1.28.5
master-2  Ready     master  74m  v1.28.5
worker-0  Ready     worker  11m  v1.28.5
worker-1  Ready     worker  11m  v1.28.5
```

NOTE

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

Additional information

- For more information on CSRs, see Certificate Signing Requests.

4.24. VERIFYING A SUCCESSFUL INSTALLATION

Verify that the OpenShift Container Platform installation is complete.

Prerequisites

- You have the installation program (**openshift-install**)

Procedure

- On a command line, enter:

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

The program outputs the console URL, as well as the administrator's login information.

4.25. TELEMETRY ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered 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.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

4.26. NEXT STEPS

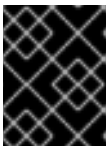
- Customize your cluster.
- Remote health reporting
- configure ingress cluster traffic by using a node port

CHAPTER 5. INSTALLING A CLUSTER ON OPENSTACK IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.15, you can install a cluster on Red Hat OpenStack Platform (RHOSP) in a restricted network by creating an internal mirror of the installation release content.

5.1. PREREQUISITES

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You read the documentation on selecting a cluster installation method and preparing it for users.
- You verified that OpenShift Container Platform 4.15 is compatible with your RHOSP version by using the Supported platforms for OpenShift clusters section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
- You created a registry on your mirror host and obtained the **imageContentSources** data for your version of OpenShift Container Platform.



IMPORTANT

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

- You understand performance and scalability practices for cluster scaling, control plane sizing, and etcd. For more information, see Recommended practices for scaling the cluster .
- You have the metadata service enabled in RHOSP.

5.2. ABOUT INSTALLATIONS IN RESTRICTED NETWORKS

In OpenShift Container Platform 4.15, 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.

5.2.1. 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.

5.3. RESOURCE GUIDELINES FOR INSTALLING OPENSHIFT CONTAINER PLATFORM ON RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 5.1. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

Resource	Value
Floating IP addresses	3
Ports	15
Routers	1
Subnets	1
RAM	88 GB
vCPUs	22
Volume storage	275 GB
Instances	7
Security groups	3
Security group rules	60
Server groups	2 - plus 1 for each additional availability zone in each machine pool

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the **swiftoperator** role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

5.3.1. Control plane machines

By default, the OpenShift Container Platform installation process creates three control plane machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs
- At least 100 GB storage space from the RHOSP quota

5.3.2. Compute machines

By default, the OpenShift Container Platform installation process creates three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 8 GB memory and 2 vCPUs
- At least 100 GB storage space from the RHOSP quota

TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

5.3.3. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory and 4 vCPUs

- At least 100 GB storage space from the RHOSP quota

5.4. INTERNET ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, you require access to the internet to obtain the images that are necessary to install your cluster.

You must have internet access to:

- 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.

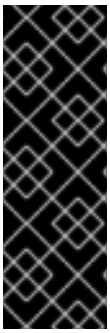
5.5. ENABLING SWIFT ON RHOSP

Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.

IMPORTANT

If the Red Hat OpenStack Platform (RHOSP) object storage service , commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.



IMPORTANT

RHOSP 17 sets the **rgw_max_attr_size** parameter of Ceph RGW to 256 characters. This setting causes issues with uploading container images to the OpenShift Container Platform registry. You must set the value of **rgw_max_attr_size** to at least 1024 characters.

Before installation, check if your RHOSP deployment is affected by this problem. If it is, reconfigure Ceph RGW.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On Ceph RGW, the **account in url** option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

```
$ openstack role add --user <user> --project <project> swiftoperator
```

Your RHOSP deployment can now use Swift for the image registry.

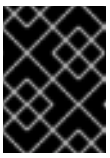
5.6. DEFINING PARAMETERS FOR THE INSTALLATION PROGRAM

The OpenShift Container Platform installation program relies on a file that is called **clouds.yaml**. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the **clouds.yaml** file:

   - If your RHOSP distribution includes the Horizon web UI, generate a **clouds.yaml** file in it.



IMPORTANT

Remember to add a password to the **auth** field. You can also keep secrets in a separate file from **clouds.yaml**.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about **clouds.yaml**, see Config files in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:

   a. Copy the certificate authority file to your machine.

   b. Add the **cacerts** key to the **clouds.yaml** file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
```

```
shiftstack:
  ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the **ca-cert.pem** key in the **cloud-provider-config** keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the **clouds.yaml** file in one of the following locations:
   a. The value of the **OS_CLIENT_CONFIG_FILE** environment variable
   b. The current directory
   c. A Unix-specific user configuration directory, for example **~/.config/openstack/clouds.yaml**
   d. A Unix-specific site configuration directory, for example **/etc/openstack/clouds.yaml**
   The installation program searches for **clouds.yaml** in that order.

5.7. SETTING OPENSTACK CLOUD CONTROLLER MANAGER OPTIONS

Optionally, you can edit the OpenStack Cloud Controller Manager (CCM) configuration for your cluster. This configuration controls how OpenShift Container Platform interacts with Red Hat OpenStack Platform (RHOSP).

For a complete list of configuration parameters, see the "OpenStack Cloud Controller Manager reference guide" page in the "Installing on OpenStack" documentation.

Procedure

1. If you have not already generated manifest files for your cluster, generate them by running the following command:

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

2. In a text editor, open the cloud-provider configuration manifest file. For example:

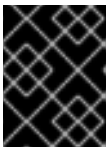
```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. Modify the options according to the CCM reference guide.
   Configuring Octavia for load balancing is a common case. For example:

```
#...
[LoadBalancer]
lb-provider = "amphora" 1
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 2
create-monitor = True 3
```

```
monitor-delay = 10s 4
monitor-timeout = 10s 5
monitor-max-retries = 1 6
#...
```

1. This property sets the Octavia provider that your load balancer uses. It accepts **"ovn"** or **"amphora"** as values. If you choose to use OVN, you must also set **lb-method** to **SOURCE_IP_PORT**.
2. This property is required if you want to use multiple external networks with your cluster. The cloud provider creates floating IP addresses on the network that is specified here.
3. This property controls whether the cloud provider creates health monitors for Octavia load balancers. Set the value to **True** to create health monitors. As of RHOSP 16.2, this feature is only available for the Amphora provider.
4. This property sets the frequency with which endpoints are monitored. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
5. This property sets the time that monitoring requests are open before timing out. The value must be in the **time.ParseDuration()** format. This property is required if the value of the **create-monitor** property is **True**.
6. This property defines how many successful monitoring requests are required before a load balancer is marked as online. The value must be an integer. This property is required if the value of the **create-monitor** property is **True**.



IMPORTANT

Prior to saving your changes, verify that the file is structured correctly. Clusters might fail if properties are not placed in the appropriate section.

IMPORTANT

You must set the value of the **create-monitor** property to **True** if you use services that have the value of the **.spec.externalTrafficPolicy** property set to **Local**. The OVN Octavia provider in RHOSP 16.2 does not support health monitors. Therefore, services that have **ETP** parameter values set to **Local** might not respond when the **lb-provider** value is set to **"ovn"**.

4. Save the changes to the file and proceed with installation.

TIP

You can update your cloud provider configuration after you run the installer. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

After you save your changes, your cluster will take some time to reconfigure itself. The process is complete if none of your nodes have a **SchedulingDisabled** status.

5.8. CREATING THE RHCOS IMAGE FOR RESTRICTED NETWORK INSTALLATIONS

Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network Red Hat OpenStack Platform (RHOSP) environment.

Prerequisites

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

Procedure

1. Log in to the Red Hat Customer Portal's Product Downloads page.
2. Under **Version**, select the most recent release of OpenShift Container Platform 4.15 for RHEL 8.

IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)** image.
4. Decompress the image.

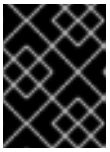
NOTE

You must decompress the image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like **.gz** or **.tgz**. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. Upload the image that you decompressed to a location that is accessible from the bastion server, like Glance. For example:

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --
disk-format qcow2 rhcos-${RHCOS_VERSION}
```



IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either **.raw** or **.qcow2** formats. If you use Ceph, you must use the **.raw** format.

WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

5.9. CREATING THE INSTALLATION CONFIGURATION FILE

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

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 **imageContentSources** values that were generated during mirror registry creation.
- 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.

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> 1
   ```

   1 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 **openstack** as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane nodes and 8 GB RAM for compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

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

```
platform:
  openstack:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
openstack.x86_64.qcow2.gz?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

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-----
```

```
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
-----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 **imageContentSources** that you recorded during mirror registry creation.

d. Optional: Set the publishing strategy to **Internal**:

```
publish: Internal
```

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

4. Make any other modifications to the **install-config.yaml** file that you require.
For more information about the parameters, see "Installation configuration parameters".

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 OpenStack

5.9.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You 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> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. 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.
4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
5. Optional: The policy to determine 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**.

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.

5.9.2. Sample customized install-config.yaml file for restricted OpenStack installations

This sample **install-config.yaml** demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

IMPORTANT

This sample file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program.

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
```

```
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle: |

  -----BEGIN CERTIFICATE-----

  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ

  -----END CERTIFICATE-----

imageContentSources:
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

5.10. GENERATING A KEY PAIR FOR CLUSTER NODE SSH ACCESS

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

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

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

IMPORTANT

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

NOTE

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

Procedure

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

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 Specify 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.

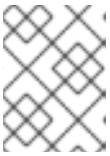
2. View the public SSH key:

```
$ 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> 1
```

1 Specify 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.

5.11. ENABLING ACCESS TO THE ENVIRONMENT

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

5.11.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

NOTE

If you do not control the DNS server, you can access the cluster by adding the cluster domain names such as the following to your **/etc/hosts** file:

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

The cluster domain names in the **/etc/hosts** file grant access to the web console and the monitoring interface of your cluster locally. You can also use the **kubectl** or **oc**. You can access the user applications by using the additional entries pointing to the <application_floating_ip>. This action makes the API and applications accessible to only you, which is not suitable for production deployment, but does allow installation for development and testing.

4. Add the FIPs to the **install-config.yaml** file as the values of the following parameters:
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.apiFloatingIP**

If you use these values, you must also enter an external network as the value of the **platform.openstack.externalNetwork** parameter in the **install-config.yaml** file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

5.11.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

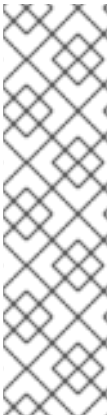
In the **install-config.yaml** file, do not define the following parameters:

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.apiFloatingIP**

If you cannot provide an external network, you can also leave **platform.openstack.externalNetwork**

blank. If you do not provide a value for **platform.openstack.externalNetwork**, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.



NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your **/etc/hosts** file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

5.12. 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

- Change to the directory that contains the installation program and initialize the cluster deployment:

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

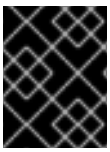
1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

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**.

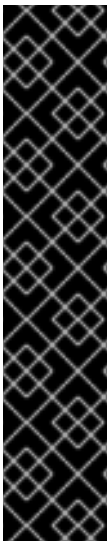


IMPORTANT

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.

5.13. VERIFYING CLUSTER STATUS

You can verify your OpenShift Container Platform cluster's status during or after installation.

Procedure

1. In the cluster environment, export the administrator's kubeconfig file:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

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

The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

```
$ oc get nodes
```

3. View your cluster's version:

```
$ oc get clusterversion
```

4. View your Operators' status:

```
$ oc get clusteroperator
```

5. View all running pods in the cluster:

```
$ oc get pods -A
```

5.14. LOGGING IN TO THE CLUSTER BY USING THE CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

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

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

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

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

5.15. 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.

5.16. TELEMETRY ACCESS FOR OPENSHIFT CONTAINER PLATFORM

In OpenShift Container Platform 4.15, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered 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.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

5.17. NEXT STEPS

- Customize your cluster.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can Remote health reporting.
- If necessary, see Registering your disconnected cluster
- Configure image streams for the Cluster Samples Operator and the **must-gather** tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks .

- If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses .

CHAPTER 6. CONFIGURING NETWORK SETTINGS AFTER INSTALLING OPENSTACK

You can configure network settings for an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) cluster after installation.

6.1. CONFIGURING APPLICATION ACCESS WITH FLOATING IP ADDRESSES

After you install OpenShift Container Platform, configure Red Hat OpenStack Platform (RHOSP) to allow application network traffic.

NOTE

You do not need to perform this procedure if you provided values for **platform.openstack.apiFloatingIP** and **platform.openstack.ingressFloatingIP** in the **install-config.yaml** file, or **os_api_fip** and **os_ingress_fip** in the **inventory.yaml** playbook, during installation. The floating IP addresses are already set.

Prerequisites

- OpenShift Container Platform cluster must be installed
- Floating IP addresses are enabled as described in the OpenShift Container Platform on RHOSP installation documentation.

Procedure

After you install the OpenShift Container Platform cluster, attach a floating IP address to the ingress port:

1. Show the port:

```
$ openstack port show <cluster_name>-<cluster_ID>-ingress-port
```

2. Attach the port to the IP address:

```
$ openstack floating ip set --port <ingress_port_ID> <apps_FIP>
```

3. Add a wildcard **A** record for ***apps.** to your DNS file:

```
*.apps.<cluster_name>.<base_domain>  IN  A  <apps_FIP>
```

NOTE

If you do not control the DNS server but want to enable application access for non-production purposes, you can add these hostnames to **/etc/hosts**:

```
<apps_FIP> console-openshift-console.apps.<cluster name>.<base domain>
<apps_FIP> integrated-oauth-server-openshift-authentication.apps.<cluster name>.
<base domain>
<apps_FIP> oauth-openshift.apps.<cluster name>.<base domain>
<apps_FIP> prometheus-k8s-openshift-monitoring.apps.<cluster name>.<base
domain>
<apps_FIP> <app name>.apps.<cluster name>.<base domain>
```

6.2. ENABLING OVS HARDWARE OFFLOADING

For clusters that run on Red Hat OpenStack Platform (RHOSP), you can enable Open vSwitch (OVS) hardware offloading.

OVS is a multi-layer virtual switch that enables large-scale, multi-server network virtualization.

Prerequisites

- You installed a cluster on RHOSP that is configured for single-root input/output virtualization (SR-IOV).
- You installed the SR-IOV Network Operator on your cluster.
- You created two **hw-offload** type virtual function (VF) interfaces on your cluster.

NOTE

Application layer gateway flows are broken in OpenShift Container Platform version 4.10, 4.11, and 4.12. Also, you cannot offload the application layer gateway flow for OpenShift Container Platform version 4.13.

Procedure

1. Create an **SriovNetworkNodePolicy** policy for the two **hw-offload** type VF interfaces that are on your cluster:

   #### The first virtual function interface

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy (1)
metadata:
  name: "hwoffload9"
  namespace: openshift-sriov-network-operator
spec:
  deviceType: netdevice
  isRdma: true
  nicSelector:
    pfNames: (2)
    - ens6
  nodeSelector:
```

```
    feature.node.kubernetes.io/network-sriov.capable: 'true'
  numVfs: 1
  priority: 99
  resourceName: "hwoffload9"
```

1. Insert the **SriovNetworkNodePolicy** value here.
2. Both interfaces must include physical function (PF) names.

The second virtual function interface

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy (1)
metadata:
  name: "hwoffload10"
  namespace: openshift-sriov-network-operator
spec:
  deviceType: netdevice
  isRdma: true
  nicSelector:
    pfNames: (2)
    - ens5
  nodeSelector:
    feature.node.kubernetes.io/network-sriov.capable: 'true'
  numVfs: 1
  priority: 99
  resourceName: "hwoffload10"
```

1. Insert the **SriovNetworkNodePolicy** value here.
2. Both interfaces must include physical function (PF) names.

2. Create **NetworkAttachmentDefinition** resources for the two interfaces:

A NetworkAttachmentDefinition resource for the first interface

```
apiVersion: k8s.cni.cncf.io/v1
kind: NetworkAttachmentDefinition
metadata:
  annotations:
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload9
  name: hwoffload9
  namespace: default
spec:
    config: '{ "cniVersion":"0.3.1", "name":"hwoffload9","type":"host-device","device":"ens6"
    }'
```

A NetworkAttachmentDefinition resource for the second interface

```
apiVersion: k8s.cni.cncf.io/v1
kind: NetworkAttachmentDefinition
metadata:
```

```
    annotations:
      k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload10
    name: hwoffload10
    namespace: default
  spec:
     config: '{ "cniVersion":"0.3.1", "name":"hwoffload10","type":"host-device","device":"ens5"
     }'
```

3. Use the interfaces that you created with a pod. For example:

A pod that uses the two OVS offload interfaces

```
apiVersion: v1
kind: Pod
metadata:
  name: dpdk-testpmd
  namespace: default
  annotations:
    irq-load-balancing.crio.io: disable
    cpu-quota.crio.io: disable
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload9
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload10
spec:
  restartPolicy: Never
  containers:
  - name: dpdk-testpmd
    image: quay.io/krister/centos8_nfv-container-dpdk-testpmd:latest
```

6.3. ATTACHING AN OVS HARDWARE OFFLOADING NETWORK

You can attach an Open vSwitch (OVS) hardware offloading network to your cluster.

Prerequisites

- Your cluster is installed and running.
- You provisioned an OVS hardware offloading network on Red Hat OpenStack Platform (RHOSP) to use with your cluster.

Procedure

1. Create a file named **network.yaml** from the following template:

```
spec:
  additionalNetworks:
  - name: hwoffload1
    namespace: cnf
    rawCNIConfig: '{ "cniVersion": "0.3.1", "name": "hwoffload1", "type": "host-
device","pciBusId": "0000:00:05.0", "ipam": {}}' 1
    type: Raw
```

where:

pciBusId

Specifies the device that is connected to the offloading network. If you do not have it, you can find this value by running the following command:

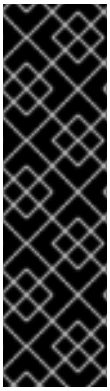
```
$ oc describe SriovNetworkNodeState -n openshift-sriov-network-operator
```

2. From a command line, enter the following command to patch your cluster with the file:

```
$ oc apply -f network.yaml
```

6.4. ENABLING IPV6 CONNECTIVITY TO PODS ON RHOSP

To enable IPv6 connectivity between pods that have additional networks that are on different nodes, disable port security for the IPv6 port of the server. Disabling port security obviates the need to create allowed address pairs for each IPv6 address that is assigned to pods and enables traffic on the security group.



IMPORTANT

Only the following IPv6 additional network configurations are supported:

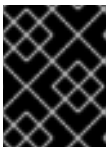
- SLAAC and host-device
- SLAAC and MACVLAN
- DHCP stateless and host-device
- DHCP stateless and MACVLAN

Procedure

- On a command line, enter the following command:

```
$ openstack port set --no-security-group --disable-port-security <compute_ipv6_port> 1
```

1 1 Specify the IPv6 port of the compute server.



IMPORTANT

This command removes security groups from the port and disables port security. Traffic restrictions are removed entirely from the port.

6.5. CREATE PODS THAT HAVE IPV6 CONNECTIVITY ON RHOSP

After you enable IPv6 connectivty for pods and add it to them, create pods that have secondary IPv6 connections.

Procedure

1. Define pods that use your IPv6 namespace and the annotation **k8s.v1.cni.cncf.io/networks: <additional_network_name>**, where **<additional_network_name** is the name of the additional network. For example, as part of a **Deployment** object:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-openshift
  namespace: ipv6
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchExpressions:
            - key: app
              operator: In
              values:
              - hello-openshift
  replicas: 2
  selector:
    matchLabels:
      app: hello-openshift
  template:
    metadata:
      labels:
        app: hello-openshift
      annotations:
        k8s.v1.cni.cncf.io/networks: ipv6
    spec:
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: hello-openshift
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
            - ALL
        image: quay.io/openshift/origin-hello-openshift
        ports:
        - containerPort: 8080
```

2. Create the pod. For example, on a command line, enter the following command:

```
$ oc create -f <ipv6_enabled_resource> 1
```

1 Specify the file that contains your resource definition.

6.6. ADDING IPV6 CONNECTIVITY TO PODS ON RHOSP

After you enable IPv6 connectivity in pods, add connectivity to them by using a Container Network Interface (CNI) configuration.

Procedure

1. To edit the Cluster Network Operator (CNO), enter the following command:

```
$ oc edit networks.operator.openshift.io cluster
```

2. Specify your CNI configuration under the **spec** field. For example, the following configuration uses a SLAAC address mode with MACVLAN:

```
...
spec:
  additionalNetworks:
  - name: ipv6
    namespace: ipv6 1
    rawCNIConfig: '{ "cniVersion": "0.3.1", "name": "ipv6", "type": "macvlan", "master": "ens4"}'
2
    type: Raw
```

1. Be sure to create pods in the same namespace.
2. The interface in the network attachment **"master"** field can differ from **"ens4"** when more networks are configured or when a different kernel driver is used.

NOTE

If you are using stateful address mode, include the IP Address Management (IPAM) in the CNI configuration.

DHCPv6 is not supported by Multus.

3. Save your changes and quit the text editor to commit your changes.

Verification

- On a command line, enter the following command:

```
$ oc get network-attachment-definitions -A
```

Example output

```
NAMESPACE      NAME          AGE
ipv6           ipv6          21h
```

You can now create pods that have secondary IPv6 connections.

CHAPTER 7. OPENSTACK CLOUD CONTROLLER MANAGER REFERENCE GUIDE

7.1. THE OPENSTACK CLOUD CONTROLLER MANAGER

Beginning with OpenShift Container Platform 4.12, clusters that run on Red Hat OpenStack Platform (RHOSP) were switched from the legacy OpenStack cloud provider to the external OpenStack Cloud Controller Manager (CCM). This change follows the move in Kubernetes from in-tree, legacy cloud providers to external cloud providers that are implemented by using the Cloud Controller Manager.

To preserve user-defined configurations for the legacy cloud provider, existing configurations are mapped to new ones as part of the migration process. It searches for a configuration called **cloud-provider-config** in the **openshift-config** namespace.

NOTE

The config map name **cloud-provider-config** is not statically configured. It is derived from the **spec.cloudConfig.name** value in the **infrastructure/cluster** CRD.

Found configurations are synchronized to the **cloud-conf** config map in the **openshift-cloud-controller-manager** namespace.

As part of this synchronization, the OpenStack CCM Operator alters the new config map such that its properties are compatible with the external cloud provider. The file is changed in the following ways:

- The **[Global] secret-name**, **[Global] secret-namespace**, and **[Global] kubeconfig-path** options are removed. They do not apply to the external cloud provider.
- The **[Global] use-clouds**, **[Global] clouds-file**, and **[Global] cloud** options are added.
- The entire **[BlockStorage]** section is removed. External cloud providers no longer perform storage operations. Block storage configuration is managed by the Cinder CSI driver.

Additionally, the CCM Operator enforces a number of default options. Values for these options are always overriden as follows:

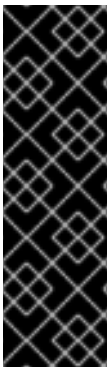
```
[Global]
use-clouds = true
clouds-file = /etc/openstack/secret/clouds.yaml
cloud = openstack
...

[LoadBalancer]
enabled = true
```

The **clouds-value** value, **/etc/openstack/secret/clouds.yaml**, is mapped to the **openstack-cloud-credentials** config in the **openshift-cloud-controller-manager** namespace. You can modify the RHOSP cloud in this file as you do any other **clouds.yaml** file.

7.2. THE OPENSTACK CLOUD CONTROLLER MANAGER (CCM) CONFIG MAP

An OpenStack CCM config map defines how your cluster interacts with your RHOSP cloud. By default, this configuration is stored under the **cloud.conf** key in the **cloud-conf** config map in the **openshift-cloud-controller-manager** namespace.



IMPORTANT

The **cloud-conf** config map is generated from the **cloud-provider-config** config map in the **openshift-config** namespace.

To change the settings that are described by the **cloud-conf** config map, modify the **cloud-provider-config** config map.

As part of this synchronization, the CCM Operator overrides some options. For more information, see "The RHOSP Cloud Controller Manager".

For example:

An example cloud-conf config map

```
apiVersion: v1
data:
  cloud.conf: |
    [Global] 1
    secret-name = openstack-credentials
    secret-namespace = kube-system
    region = regionOne
    [LoadBalancer]
    enabled = True
kind: ConfigMap
metadata:
  creationTimestamp: "2022-12-20T17:01:08Z"
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
  resourceVersion: "2519"
  uid: cbbeedaf-41ed-41c2-9f37-4885732d3677
```

1 Set global options by using a **clouds.yaml** file rather than modifying the config map.

The following options are present in the config map. Except when indicated otherwise, they are mandatory for clusters that run on RHOSP.

7.2.1. Load balancer options

CCM supports several load balancer options for deployments that use Octavia.

NOTE

Neutron-LBaaS support is deprecated.

Option	Description
enabled	Whether or not to enable the **LoadBalancer** type of services integration. The default value is **true**.
floating-network-id	Optional. The external network used to create floating IP addresses for load balancer virtual IP addresses (VIPs). If there are multiple external networks in the cloud, this option must be set or the user must specify **loadbalancer.openstack.org/floating-network-id** in the service annotation.
floating-subnet-id	Optional. The external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation **loadbalancer.openstack.org/floating-subnet-id**.
floating-subnet	Optional. A name pattern (glob or regular expression if starting with ~) for the external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation **loadbalancer.openstack.org/floating-subnet**. If multiple subnets match the pattern, the first one with available IP addresses is used.
floating-subnet-tags	Optional. Tags for the external network subnet used to create floating IP addresses for the load balancer VIP. Can be overridden by the service annotation **loadbalancer.openstack.org/floating-subnet-tags**. If multiple subnets match these tags, the first one with available IP addresses is used. If the RHOSP network is configured with sharing disabled, for example, with the **--no-share** flag used during creation, this option is unsupported. Set the network to share to use this option.

Option	Description
lb-method	The load balancing algorithm used to create the load balancer pool. For the Amphora provider the value can be **ROUND_ROBIN**, **LEAST_CONNECTIONS**, or **SOURCE_IP**. The default value is **ROUND_ROBIN**. For the OVN provider, only the **SOURCE_IP_PORT** algorithm is supported. For the Amphora provider, if using the **LEAST_CONNECTIONS** or **SOURCE_IP** methods, configure the **create-monitor** option as **true** in the **cloud-provider-config** config map on the **openshift-config** namespace and **ETP:Local** on the load-balancer type service to allow balancing algorithm enforcement in the client to service endpoint connections.
lb-provider	Optional. Used to specify the provider of the load balancer, for example, **amphora** or **octavia**. Only the Amphora and Octavia providers are supported.
lb-version	Optional. The load balancer API version. Only **"v2"** is supported.
subnet-id	The ID of the Networking service subnet on which load balancer VIPs are created. For dual stack deployments, leave this option unset. The OpenStack cloud provider automatically selects which subnet to use for a load balancer.
network-id	The ID of the Networking service network on which load balancer VIPs are created. Unnecessary if **subnet-id** is set. If this property is not set, the network is automatically selected based on the network that cluster nodes use.
create-monitor	Whether or not to create a health monitor for the service load balancer. A health monitor is required for services that declare **externalTrafficPolicy: Local**. The default value is **false**. This option is unsupported if you use RHOSP earlier than version 17 with the **ovn** provider.
monitor-delay	The interval in seconds by which probes are sent to members of the load balancer. The default value is **5**.

Option	Description
monitor-max-retries	The number of successful checks that are required to change the operating status of a load balancer member to **ONLINE**. The valid range is **1** to **10**, and the default value is **1**.
monitor-timeout	The time in seconds that a monitor waits to connect to the back end before it times out. The default value is **3**.
internal-lb	Whether or not to create an internal load balancer without floating IP addresses. The default value is **false**.
LoadBalancerClass "ClassName"	This is a config section that comprises a set of options: • **floating-network-id** • **floating-subnet-id** • **floating-subnet** • **floating-subnet-tags** • **network-id** • **subnet-id** The behavior of these options is the same as that of the identically named options in the load balancer section of the CCM config file. You can set the **ClassName** value by specifying the service annotation **loadbalancer.openstack.org/class**.
max-shared-lb	The maximum number of services that can share a load balancer. The default value is **2**.

7.2.2. Options that the Operator overrides

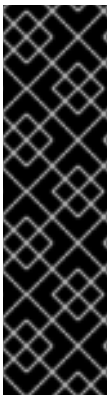
The CCM Operator overrides the following options, which you might recognize from configuring RHOSP. Do not configure them yourself. They are included in this document for informational purposes only.

Option	Description
auth-url	The RHOSP Identity service URL. For example, **http://128.110.154.166/identity**.

Option	Description
os-endpoint-type	The type of endpoint to use from the service catalog.
username	The Identity service user name.
password	The Identity service user password.
domain-id	The Identity service user domain ID.
domain-name	The Identity service user domain name.
tenant-id	The Identity service project ID. Leave this option unset if you are using Identity service application credentials. In version 3 of the Identity API, which changed the identifier **tenant** to **project**, the value of **tenant-id** is automatically mapped to the project construct in the API.
tenant-name	The Identity service project name.
tenant-domain-id	The Identity service project domain ID.
tenant-domain-name	The Identity service project domain name.
user-domain-id	The Identity service user domain ID.
user-domain-name	The Identity service user domain name.
use-clouds	Whether or not to fetch authorization credentials from a **clouds.yaml** file. Options set in this section are prioritized over values read from the **clouds.yaml** file. CCM searches for the file in the following places: 1. The value of the **clouds-file** option. 2. A file path stored in the environment variable **OS_CLIENT_CONFIG_FILE**. 3. The directory **pkg/openstack**. 4. The directory **~/.config/openstack**. 5. The directory **/etc/openstack**.

Option	Description
clouds-file	The file path of a **clouds.yaml** file. It is used if the **use-clouds** option is set to **true**.
cloud	The named cloud in the **clouds.yaml** file that you want to use. It is used if the **use-clouds** option is set to **true**.

CHAPTER 8. DEPLOYING ON OPENSTACK WITH ROOTVOLUME AND ETCD ON LOCAL DISK



IMPORTANT

Deploying on Red Hat OpenStack Platform (RHOSP) with rootVolume and etcd on local disk 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 .

As a day 2 operation, you can resolve and prevent performance issues of your Red Hat OpenStack Platform (RHOSP) installation by moving etcd from a root volume (provided by OpenStack Cinder) to a dedicated ephemeral local disk.

8.1. DEPLOYING RHOSP ON LOCAL DISK

If you have an existing RHOSP cloud, you can move etcd from that cloud to a dedicated ephemeral local disk.

WARNING

This procedure is for testing etcd on a local disk only and should not be used on production clusters. In certain cases, complete loss of the control plane can occur. For more information, see "Overview of backup and restore operation" under "Backup and restore".

Prerequisites

- You have an OpenStack cloud with a working Cinder.
- Your OpenStack cloud has at least 75 GB of available storage to accommodate 3 root volumes for the OpenShift control plane.
- The OpenStack cloud is deployed with Nova ephemeral storage that uses a local storage backend and not **rbd**.

Procedure

1. Create a Nova flavor for the control plane with at least 10 GB of ephemeral disk by running the following command, replacing the values for **--ram**, **--disk**, and <flavor_name> based on your environment:

```
$ openstack flavor create --<ram 16384> --<disk 0> --ephemeral 10 --vcpus 4
<flavor_name>
```

2. Deploy a cluster with root volumes for the control plane; for example:

Example YAML file

```
# ...
controlPlane:
  name: master
  platform:
    openstack:
      type: ${CONTROL_PLANE_FLAVOR}
      rootVolume:
        size: 25
        types:
        - ${CINDER_TYPE}
  replicas: 3
# ...
```

3. Deploy the cluster you created by running the following command:

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

1 For **<installation_directory>**, specify the location of the customized **./install-config.yaml file** that you previously created.

4. Verify that the cluster you deployed is healthy before proceeding to the next step by running the following command:

```
$ oc wait clusteroperators --all --for=condition=Progressing=false 1
```

1 Ensures that the cluster operators are finished progressing and that the cluster is not deploying or updating.

5. Edit the **ControlPlaneMachineSet** (CPMS) to add the additional block ephemeral device that is used by etcd by running the following command:

```
$ oc patch ControlPlaneMachineSet/cluster -n openshift-machine-api --type json -p ' 1
[
  {
    "op": "add",
    "path":
"/spec/template/machines_v1beta1_machine_openshift_io/spec/providerSpec/value/additionalB
ockDevices", 2
    "value": [
      {
        "name": "etcd",
        "sizeGiB": 10,
        "storage": {
          "type": "Local" 3
        }
```

```
        }
      ]
    }
  ]
'
```

1. Applies the JSON patch to the **ControlPlaneMachineSet** custom resource (CR).
2. Specifies the path where the **additionalBlockDevices** are added.
3. Adds the etcd devices with at least local storage of 10 GB to the cluster. You can specify values greater than 10 GB as long as the etcd device fits the Nova flavor. For example, if the Nova flavor has 15 GB, you can create the etcd device with 12 GB.

6. Verify that the control plane machines are healthy by using the following steps:
   a. Wait for the control plane machine set update to finish by running the following command:

```
$ oc wait --timeout=90m --for=condition=Progressing=false
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

   b. Verify that the 3 control plane machine sets are updated by running the following command:

```
$ oc wait --timeout=90m --for=jsonpath='{.status.updatedReplicas}'=3
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

   c. Verify that the 3 control plane machine sets are healthy by running the following command:

```
$ oc wait --timeout=90m --for=jsonpath='{.status.replicas}'=3
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

   d. Verify that the **ClusterOperators** are not progressing in the cluster by running the following command:

```
$ oc wait clusteroperators --timeout=30m --all --for=condition=Progressing=false
```

   e. Verify that each of the 3 control plane machines has the additional block device you previously created by running the following script:

```
$ cp_machines=$(oc get machines -n openshift-machine-api --
selector='machine.openshift.io/cluster-api-machine-role=master' --no-headers -o custom-
columns=NAME:.metadata.name) 1


if [[ $(echo "${cp_machines}" | wc -l) -ne 3 ]]; then
  exit 1
fi 2


for machine in ${cp_machines}; do
  if ! oc get machine -n openshift-machine-api "${machine}" -o
jsonpath='{.spec.providerSpec.value.additionalBlockDevices}' | grep -q 'etcd'; then
```

```
    exit 1
  fi 3
done
```

1. Retrieves the control plane machines running in the cluster.
2. Iterates over machines which have an **additionalBlockDevices** entry with the name **etcd**.
3. Outputs the name of every control plane machine which has an **additionalBlockDevice** named **etcd**.

7. Create a file named **98-var-lib-etcd.yaml** by using the following YAML file:

WARNING

This procedure is for testing etcd on a local disk and should not be used on a production cluster. In certain cases, complete loss of the control plane can occur. For more information, see "Overview of backup and restore operation" under "Backup and restore".

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 98-var-lib-etcd
spec:
  config:
    ignition:
      version: 3.4.0
    systemd:
      units:
      - contents: |
          [Unit]
          Description=Mount local-etcd to /var/lib/etcd

          [Mount]
          What=/dev/disk/by-label/local-etcd 1
          Where=/var/lib/etcd
          Type=xfs
          Options=defaults,prjquota

          [Install]
          WantedBy=local-fs.target
        enabled: true
        name: var-lib-etcd.mount
      - contents: |
          [Unit]
          Description=Create local-etcd filesystem
```

```
        DefaultDependencies=no
        After=local-fs-pre.target
        ConditionPathIsSymbolicLink=!/dev/disk/by-label/local-etcd 2

        [Service]
        Type=oneshot
        RemainAfterExit=yes
        ExecStart=/bin/bash -c "[ -L /dev/disk/by-label/ephemeral0 ] || ( >&2 echo Ephemeral
disk does not exist; /usr/bin/false )"
        ExecStart=/usr/sbin/mkfs.xfs -f -L local-etcd /dev/disk/by-label/ephemeral0 3

        [Install]
        RequiredBy=dev-disk-by\x2dlabel-local\x2detcd.device
      enabled: true
      name: create-local-etcd.service
    - contents: |
        [Unit]
        Description=Migrate existing data to local etcd
        After=var-lib-etcd.mount
        Before=crio.service 4

        Requisite=var-lib-etcd.mount
        ConditionPathExists=!/var/lib/etcd/member
        ConditionPathIsDirectory=/sysroot/ostree/deploy/rhcos/var/lib/etcd/member 5

        [Service]
        Type=oneshot
        RemainAfterExit=yes

        ExecStart=/bin/bash -c "if [ -d /var/lib/etcd/member.migrate ]; then rm -rf
/var/lib/etcd/member.migrate; fi" 6

        ExecStart=/usr/bin/cp -aZ /sysroot/ostree/deploy/rhcos/var/lib/etcd/member/
/var/lib/etcd/member.migrate
        ExecStart=/usr/bin/mv /var/lib/etcd/member.migrate /var/lib/etcd/member 7

        [Install]
        RequiredBy=var-lib-etcd.mount
      enabled: true
      name: migrate-to-local-etcd.service
    - contents: |
        [Unit]
        Description=Relabel /var/lib/etcd

        After=migrate-to-local-etcd.service
        Before=crio.service

        [Service]
        Type=oneshot
        RemainAfterExit=yes

        ExecCondition=/bin/bash -c "[ -n \"$(restorecon -nv /var/lib/etcd)\" ]" 8

        ExecStart=/usr/sbin/restorecon -R /var/lib/etcd
```

```
    [Install]
    RequiredBy=var-lib-etcd.mount
enabled: true
name: relabel-var-lib-etcd.service
```

1. The etcd database must be mounted by the device, not a label, to ensure that **systemd** generates the device dependency used in this config to trigger filesystem creation.
2. Do not run if the file system **dev/disk/by-label/local-etcd** already exists.
3. Fails with an alert message if **/dev/disk/by-label/ephemeral0** doesn't exist.
4. Migrates existing data to local etcd database. This config does so after **/var/lib/etcd** is mounted, but before CRI-O starts so etcd is not running yet.
5. Requires that etcd is mounted and does not contain a member directory, but the ostree does.
6. Cleans up any previous migration state.
7. Copies and moves in separate steps to ensure atomic creation of a complete member directory.
8. Performs a quick check of the mount point directory before performing a full recursive relabel. If restorecon in the file path **/var/lib/etcd** cannot rename the directory, the recursive rename is not performed.

8. Create the new **MachineConfig** object by running the following command:

```
$ oc create -f 98-var-lib-etcd.yaml
```

NOTE

Moving the etcd database onto the local disk of each control plane machine takes time.

9. Verify that the etcd databases has been transferred to the local disk of each control plane by running the following commands:
   a. Verify that the cluster is still updating by running the following command:

```
$ oc wait --timeout=45m --for=condition=Updating=false machineconfigpool/master
```

   b. Verify that the cluster is ready by running the following command:

```
$ oc wait node --selector='node-role.kubernetes.io/master' --for condition=Ready --
timeout=30s
```

   c. Verify that the cluster Operators are running in the cluster by running the following command:

```
$ oc wait clusteroperators --timeout=30m --all --for=condition=Progressing=false
```

8.2. ADDITIONAL RESOURCES

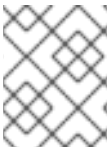
- Recommended etcd practices
- Overview of backup and restore options

CHAPTER 9. UNINSTALLING A CLUSTER ON OPENSTACK

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP).

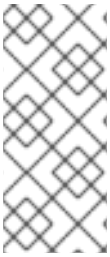
9.1. REMOVING A CLUSTER THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.



NOTE

If you deployed your cluster to the AWS C2S Secret Region, the installation program does not support destroying the cluster; you must manually remove the cluster resources.



NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access. For example, some Google Cloud resources require IAM permissions in shared VPC host projects, or there might be unused health checks that must be deleted .

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

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

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

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

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

NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

CHAPTER 10. UNINSTALLING A CLUSTER ON RHOSP FROM YOUR OWN INFRASTRUCTURE

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP) on user-provisioned infrastructure.

10.1. DOWNLOADING PLAYBOOK DEPENDENCIES

The Ansible playbooks that simplify the removal process on user-provisioned infrastructure require several Python modules. On the machine where you will run the process, add the modules' repositories and then download them.

NOTE

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:
   a. Register with Red Hat Subscription Manager:

```
$ sudo subscription-manager register # If not done already
```

   b. Pull the latest subscription data:

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. Disable the current repositories:

```
$ sudo subscription-manager repos --disable=* # If not done already
```

   d. Add the required repositories:

```
$ sudo subscription-manager repos \
  --enable=rhel-9-for-x86_64-appstream-rpms \
  --enable=rhel-9-for-x86_64-baseos-rpms \
  --enable=openstack-17.1-for-rhel-9-x86_64-rpms
```

2. Install the modules:

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk
```

3. Ensure that the **python** command points to **python3**:

```
$ sudo alternatives --set python /usr/bin/python3
```

10.2. REMOVING A CLUSTER FROM RHOSP THAT USES YOUR OWN INFRASTRUCTURE

You can remove an OpenShift Container Platform cluster on Red Hat OpenStack Platform (RHOSP) that uses your own infrastructure. To complete the removal process quickly, run several Ansible playbooks.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies."
- You have the playbooks that you used to install the cluster.
- You modified the playbooks that are prefixed with **down-** to reflect any changes that you made to their corresponding installation playbooks. For example, changes to the **bootstrap.yaml** file are reflected in the **down-bootstrap.yaml** file.
- All of the playbooks are in a common directory.

Procedure

1. On a command line, run the playbooks that you downloaded:

```
$ ansible-playbook -i inventory.yaml  \
 down-bootstrap.yaml      \
 down-control-plane.yaml  \
 down-compute-nodes.yaml  \
 down-load-balancers.yaml \
 down-network.yaml        \
 down-security-groups.yaml
```

2. Remove any DNS record changes you made for the OpenShift Container Platform installation.

OpenShift Container Platform is removed from your infrastructure.

CHAPTER 11. INSTALLATION CONFIGURATION PARAMETERS FOR OPENSTACK

Before you deploy an OpenShift Container Platform cluster on Red Hat OpenStack Platform (RHOSP), 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.

11.1. AVAILABLE INSTALLATION CONFIGURATION PARAMETERS FOR OPENSTACK

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

11.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 11.1. Required parameters

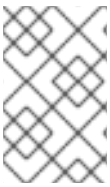
Parameter	Description	Values
apiVersion:	The API version for the **install-config.yaml** content. The current version is **v1**. The installation program might also support older API versions.	String
baseDomain:	The base domain of your cloud provider. The base domain is used to create routes to your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

Parameter	Description	Values
metadata: name:	The name of the cluster. DNS records for the cluster are all subdomains of **{{.metadata.name}}.{{.baseDomain}}**.	String of lowercase letters, hyphens (**-**), and periods (**.**), such as **dev**. The string must be 14 characters or fewer long.
platform:	The configuration for the specific platform upon which to perform the installation: **alibabacloud**, **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**. For additional information about **platform.<platform>** parameters, consult the table for your specific platform that follows.	Object
pullSecret:	Get a pull secret from Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

11.1.2. 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.



NOTE

Globalnet is not supported with Red Hat OpenShift Data Foundation disaster recovery solutions. For regional disaster recovery scenarios, ensure that you use a nonoverlapping range of private IP addresses for the cluster and service networks in each cluster.

Table 11.2. Network parameters

Parameter	Description	Values
`networking:`	The configuration for the cluster network.	Object  **NOTE** You cannot change parameters specified by the **networking** object after installation.
`networking:` `networkType:`	The Red Hat OpenShift Networking network plugin to install.	**OVNKubernetes**. **OVNKubernetes** is a Container Network Interface (CNI) plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is **OVNKubernetes**.
`networking:` `clusterNetwork:`	The IP address blocks for pods. The default value is **10.128.0.0/14** with a host prefix of **/23**. If you specify multiple IP address blocks, the blocks must not overlap.	An array of objects. For example: `networking:` `clusterNetwork:` `- cidr: 10.128.0.0/14` `hostPrefix: 23`
`networking:` `clusterNetwork:` `cidr:`	Required if you use **networking.clusterNetwork**. An IP address block. An IPv4 network.	An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between **0** and **32**.
`networking:` `clusterNetwork:` `hostPrefix:`	The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23** then each node is assigned a **/23** subnet out of the given **cidr**. A **hostPrefix** value of **23** provides 510 (2^(32 - 23) - 2) pod IP addresses.	A subnet prefix. The default value is **23**.
`networking:` `serviceNetwork:`	The IP address block for services. The default value is **172.30.0.0/16**. The OVN-Kubernetes network plugins supports only a single IP address block for the service network.	An array with an IP address block in CIDR format. For example: `networking:` `serviceNetwork:` `- 172.30.0.0/16`

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

11.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 11.3. Optional parameters

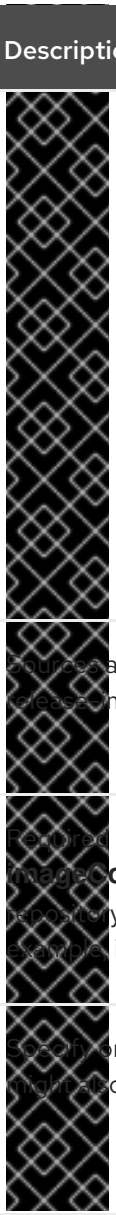
Parameter	Description	Values
additionalTrustBundle:	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle might also be used when a proxy is configured.	String
capabilities:	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in *Installing*.	String array
capabilities: baselineCapabilitySet:	Selects an initial set of optional capabilities to enable. Valid values are **None**, **v4.11**, **v4.12** and **vCurrent**. The default value is **vCurrent**.	String

Parameter	Description	Values
capabilities: additionalEnabledCapabilities:	Extends the set of optional capabilities beyond what you specify in **baselineCapabilitySet**. You can specify multiple capabilities in this parameter.	String array
cpuPartitioningMode:	Enables workload partitioning, which isolates OpenShift Container Platform services, cluster management workloads, and infrastructure pods to run on a reserved set of CPUs. You can only enable workload partitioning during installation. You cannot disable it after installation. While this field enables workload partitioning, it does not configure workloads to use specific CPUs. For more information, see the *Workload partitioning* page in the *Scalability and Performance* section.	**None** or **AllNodes**. **None** is the default value.
compute:	The configuration for the machines that comprise the compute nodes.	Array of **MachinePool** objects.
compute: architecture:	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are **amd64** and **arm64**. Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see *Supported installation methods for different platforms* in *Selecting a cluster installation method and preparing it for users*.	String

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

Parameter	Description	Values
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** and **arm64**. Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see *Supported installation methods for different platforms* in *Selecting a cluster installation method and preparing it for users*.	String
controlPlane: hyperthreading:	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  **IMPORTANT** If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
controlPlane: name:	Required if you use **controlPlane**. The name of the machine pool.	**master**
controlPlane: platform:	Required if you use **controlPlane**. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the **compute.platform** parameter value.	**alibabacloud**, **aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**

Parameter	Description	Values
controlPlane: replicas:	The number of control plane machines to provision.	Supported values are **3**, or **1** when deploying single-node OpenShift.
credentialsMode:	The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.	**Mint**, **Passthrough**, **Manual** or an empty string (**""**). [1]
fips:	Enable or disable FIPS mode. The default is **false** (disabled). If you enable FIPS mode, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that RHCOS provides instead.	**false** or **true**

Parameter	Description	Values
	 **IMPORTANT** To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode. When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 validation on only the x86_64, ppc64le, and s390x architectures. If you are using Azure File storage, you cannot enable FIPS mode.	
imageContentSources:	Sources and repositories for the release-image content.	Array of objects. Includes a **source** and, optionally, **mirrors**, as described in the following rows of this table.
imageContentSources: source:	Required if you use **imageContentSources**. Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources: mirrors:	Specify one or more repositories that might also contain the same images.	Array of strings
platform: aws: lbType:	Required to set the NLB load balancer type in AWS. Valid values are **Classic** or **NLB**. If no value is specified, the installation program defaults to **Classic**. The installation program sets the value provided here in the ingress cluster configuration object. If you do not specify a load balancer type for other Ingress Controllers, they use the type set in this parameter.	**Classic** or **NLB**. The default value is **Classic**.
publish:	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	**Internal** or **External**. To deploy a private cluster that cannot be accessed from the internet, set the **publish** parameter to **Internal**. The default value is **External**.

Parameter	Description	Values
sshKey:	The SSH key to authenticate access to your cluster machines. **NOTE** For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.	For example, **sshKey: ssh-ed25519 AAAA..**.

1. 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.

NOTE

If your AWS account has service control policies (SCP) enabled, you must configure the **credentialsMode** parameter to **Mint**, **Passthrough**, or **Manual**. If you are installing on Google Cloud into a shared virtual private cloud (VPC), **credentialsMode** must be set to **Passthrough** or **Manual**.

IMPORTANT

Setting this parameter to **Manual** enables alternatives to storing administrator-level secrets in the **kube-system** project, which require additional configuration steps. For more information, see "Alternatives to storing administrator-level secrets in the kube-system project".

11.1.4. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

Table 11.4. Optional AWS parameters

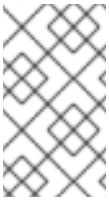
Parameter	Description	Values
compute: platform: aws: amiID:	The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See *RHCOS AMIs for AWS infrastructure* for available AMI IDs.

Parameter	Description	Values
`compute:` `  platform:` `  aws:` `  iamRole:`	A pre-existing AWS IAM role applied to the compute machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
`compute:` `  platform:` `  aws:` `rootVolume:` `  iops:`	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.	Integer, for example **4000**.
`compute:` `  platform:` `  aws:` `rootVolume:` `  size:`	The size in GiB of the root volume.	Integer, for example **500**.
`compute:` `  platform:` `  aws:` `rootVolume:` `  type:`	The type of the root volume.	Valid AWS EBS volume type, such as **io1**.
`compute:` `  platform:` `  aws:` `rootVolume:` `kmsKeyARN:`	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of worker nodes with a specific KMS key.	Valid key ID or the key ARN

Parameter	Description	Values
`compute:` `  platform:` `  aws:` `  type:`	The EC2 instance type for the compute machines.	Valid AWS instance type, such as **m4.2xlarge**. See the "Tested instance types for AWS" table on the "Installing a cluster on AWS with customizations" page.
`compute:` `  platform:` `  aws:` `  zones:`	The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.	A list of valid AWS availability zones, such as **us-east-1c**, in a YAML sequence.
`controlPlane:` `  platform:` `  aws:` `  amiID:`	The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See *RHCOS AMIs for AWS infrastructure* for available AMI IDs.
`controlPlane:` `  platform:` `  aws:` `  iamRole:`	A pre-existing AWS IAM role applied to the control plane machine pool instance profiles. You can use these fields to match naming schemes and include predefined permissions boundaries for your IAM roles. If undefined, the installation program creates a new IAM role.	The name of a valid AWS IAM role.
`controlPlane:` `  platform:` `  aws:` `rootVolume:` `  iops:`	The Input/Output Operations Per Second (IOPS) that is reserved for the root volume on control plane machines.	Integer, for example **4000**.
`controlPlane:` `  platform:` `  aws:` `rootVolume:` `  size:`	The size in GiB of the root volume for control plane machines.	Integer, for example **500**.

Parameter	Description	Values
controlPlane: platform: aws: rootVolume: type:	The type of the root volume for control plane machines.	Valid AWS EBS volume type, such as **io1**.
controlPlane: platform: aws: rootVolume: kmsKeyARN:	The Amazon Resource Name (key ARN) of a KMS key. This is required to encrypt operating system volumes of control plane nodes with a specific KMS key.	Valid key ID and the key ARN
controlPlane: platform: aws: type:	The EC2 instance type for the control plane machines.	Valid AWS instance type, such as **m6i.xlarge**. See the "Tested instance types for AWS" table on the "Installing a cluster on AWS with customizations" page.
controlPlane: platform: aws: zones:	The availability zones where the installation program creates machines for the control plane machine pool.	A list of valid AWS availability zones, such as **us-east-1c**, in a YAML sequence.
platform: aws: amiID:	The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.	Any published or custom RHCOS AMI that belongs to the set AWS region. See *RHCOS AMIs for AWS infrastructure* for available AMI IDs.

Parameter	Description	Values
platform: aws: hostedZone:	An existing Route 53 private hosted zone for the cluster. You can only use a pre-existing hosted zone when also supplying your own VPC. The hosted zone must already be associated with the user-provided VPC before installation. Also, the domain of the hosted zone must be the cluster domain or a parent of the cluster domain. If undefined, the installation program creates a new hosted zone.	String, for example **Z3URY6TWQ91KVV**.
platform: aws: hostedZoneRole:	An Amazon Resource Name (ARN) for an existing IAM role in the account containing the specified hosted zone. The installation program and cluster operators assume this role when performing operations on the hosted zone. Use this parameter only when you are installing a cluster into a shared VPC.	String, for example **arn:aws:iam::1234567890:role/shared-vpc-role**.
platform: aws: region:	The AWS region that the installation program creates all cluster resources in.	Any valid AWS region, such as **us-east-1**. You can use the AWS CLI to access the regions available based on your selected instance type by running the following command: `$ aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge` **IMPORTANT** When running on ARM based AWS instances, ensure that you enter a region where AWS Graviton processors are available. See Global availability map in the AWS documentation. Currently, AWS Graviton3 processors are only available in some regions.

Parameter	Description	Values
platform: aws: serviceEndpoi nts: - name: url:	The AWS service endpoint name and URL. Custom endpoints are only required for cases where alternative AWS endpoints, such as FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.	Valid AWS service endpoint name and valid AWS service endpoint URL.
platform: aws: userTags:	A map of keys and values that the installation program adds as tags to all resources that it creates.	Any valid YAML map, such as key value pairs in the **<key>: <value>** format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.  **NOTE** You can add up to 25 user-defined tags during installation. The remaining 25 tags are reserved for OpenShift Container Platform.
platform: aws: propagateUse rTags:	A flag that directs in-cluster Operators to include the specified user tags in the tags of the AWS resources that the Operators create.	Boolean values, for example **true** or **false**.

Parameter	Description	Values
platform: aws: subnets:	If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same **machineNetwork[].cidr** ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone. For clusters that use AWS Local Zones, you must add AWS Local Zone subnets to this list to ensure edge machine pool creation.	Valid subnet IDs.
platform: aws: preserveBootstrapIgnition:	Prevents the S3 bucket from being deleted after completion of bootstrapping.	**true** or **false**. The default value is **false**, which results in the S3 bucket being deleted.

11.1.5. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 11.5. Additional RHOSP parameters

Parameter	Description	Values
compute: platform: openstack: rootVolume: size:	For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example **30**.

Parameter	Description	Values
compute: platform: openstack: rootVolume: types:	For compute machines, the root volume types.	A list of strings, for example, {**performance-host1**, **performance-host2**, **performance-host3**}. [1]
compute: platform: openstack: rootVolume: type:	For compute machines, the root volume's type. This property is deprecated and is replaced by **compute.platform.openstack.rootVolume.types**.	String, for example, **performance**. [2]
compute: platform: openstack: rootVolume: zones:	For compute machines, the Cinder availability zone to install root volumes on. If you do not set a value for this parameter, the installation program selects the default availability zone. This parameter is mandatory when **compute.platform.openstack.zones** is defined.	A list of strings, for example **["zone-1", "zone-2"]**.
controlPlane: platform: openstack: rootVolume: size:	For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.	Integer, for example **30**.
controlPlane: platform: openstack: rootVolume: types:	For control plane machines, the root volume types.	A list of strings, for example, {**performance-host1**, **performance-host2**, **performance-host3**}. [1]

Parameter	Description	Values
controlPlane: platform: openstack: rootVolume: type:	For control plane machines, the root volume's type. This property is deprecated and is replaced by **compute.platform.openstack.rootVolume.types**.	String, for example, **performance**. [2]
controlPlane: platform: openstack: rootVolume: zones:	For control plane machines, the Cinder availability zone to install root volumes on. If you do not set this value, the installation program selects the default availability zone. This parameter is mandatory when **controlPlane.platform.openstack.zones** is defined.	A list of strings, for example **["zone-1", "zone-2"]**.
platform: openstack: cloud:	The name of the RHOSP cloud to use from the list of clouds in the **clouds.yaml** file. In the cloud configuration in the **clouds.yaml** file, if possible, use application credentials rather than a user name and password combination. Using application credentials avoids disruptions from secret propogation that follow user name and password rotation.	String, for example **MyCloud**.
platform: openstack: externalNetwork:	The RHOSP external network name to be used for installation.	String, for example **external**.

Parameter	Description	Values
platform: openstack: computeFlavor:	The RHOSP flavor to use for control plane and compute machines. This property is deprecated. To use a flavor as the default for all machine pools, add it as the value of the **type** key in the **platform.openstack.defaultMachinePlatform** property. You can also set a flavor value for each machine pool individually.	String, for example **m1.xlarge**.

1. If the machine pool defines **zones**, the count of types can either be a single item or match the number of items in **zones**. For example, the count of types cannot be 2 if there are 3 items in **zones**.

2. If you have any existing reference to this property, the installer populates the corresponding value in the **controlPlane.platform.openstack.rootVolume.types** field.

11.1.6. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 11.6. Optional RHOSP parameters

Parameter	Description	Values
compute: platform: openstack: additionalNetworkIDs:	Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.	A list of one or more UUIDs as strings. For example, **fa806b2f-ac49-4bce-b9db-124bc64209bf**.
compute: platform: openstack: additionalSecurityGroupIDs:	Additional security groups that are associated with compute machines.	A list of one or more UUIDs as strings. For example, **7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**.

Parameter	Description	Values
`compute:` `  platform:` `  openstack:` `  zones:`	RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.	A list of strings. For example, **["zone-1", "zone-2"]**.
`compute:` `  platform:` `  openstack:` `serverGroupPolicy:`	The server group policy to apply to the group that contains the compute machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include **anti-affinity**, **soft-affinity**, and **soft-anti-affinity**. The default value is **soft-anti-affinity**. An **affinity** policy prevents migrations and therefore affects RHOSP upgrades. The **affinity** policy is not supported. If you use a strict **anti-affinity** policy, an additional RHOSP host is required during instance migration.	A server group policy to apply to the machine pool. For example, **soft-affinity**.
`controlPlane:` `  platform:` `  openstack:` `additionalNetworkIDs:`	Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks. Additional networks that are attached to a control plane machine are also attached to the bootstrap node.	A list of one or more UUIDs as strings. For example, **fa806b2f-ac49-4bce-b9db-124bc64209bf**.
`controlPlane:` `  platform:` `  openstack:` `additionalSecurityGroupIDs:`	Additional security groups that are associated with control plane machines.	A list of one or more UUIDs as strings. For example, **7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**.

Parameter	Description	Values
controlPlane: platform: openstack: zones:	RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installation program relies on the default settings for Nova that the RHOSP administrator configured.	A list of strings. For example, **["zone-1", "zone-2"]**.
controlPlane: platform: openstack: serverGroupPolicy:	Server group policy to apply to the group that contains the control plane machines in the pool. You cannot change server group policies or affiliations after creation. Supported options include **anti-affinity**, **soft-affinity**, and **soft-anti-affinity**. The default value is **soft-anti-affinity**. An **affinity** policy prevents migrations, and therefore affects RHOSP upgrades. The **affinity** policy is not supported. If you use a strict **anti-affinity** policy, an additional RHOSP host is required during instance migration.	A server group policy to apply to the machine pool. For example, **soft-affinity**.
platform: openstack: clusterOSImage:	The location from which the installation program downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.	An HTTP or HTTPS URL, optionally with an SHA-256 checksum. For example, **http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**. The value can also be the name of an existing Glance image, for example **my-rhcos**.

Parameter	Description	Values
`platform: openstack: clusterOSImageProperties:`	Properties to add to the installer-uploaded ClusterOSImage in Glance. This property is ignored if **platform.openstack.clusterOSImage** is set to an existing Glance image. You can use this property to exceed the default persistent volume (PV) limit for RHOSP of 26 PVs per node. To exceed the limit, set the **hw_scsi_model** property value to **virtio-scsi** and the **hw_disk_bus** value to **scsi**. You can also use this property to enable the QEMU guest agent by including the **hw_qemu_guest_agent** property with a value of **yes**.	A set of string properties. For example: `clusterOSImageProperties:` `  hw_scsi_model: "virtio-scsi"` `  hw_disk_bus: "scsi"` `  hw_qemu_guest_agent: "yes"`
`platform: openstack: defaultMachinePlatform:`	The default machine pool platform configuration.	`{` `  "type": "ml.large",` `  "rootVolume": {` `  "size": 30,` `  "type": "performance"` `  }` `}`
`platform: openstack: ingressFloatingIP:`	An existing floating IP address to associate with the Ingress port. To use this property, you must also define the **platform.openstack.externalNetwork** property.	An IP address, for example **128.0.0.1**.
`platform: openstack: apiFloatingIP:`	An existing floating IP address to associate with the API load balancer. To use this property, you must also define the **platform.openstack.externalNetwork** property.	An IP address, for example **128.0.0.1**.

Parameter	Description	Values
platform: openstack: externalDNS:	IP addresses for external DNS servers that cluster instances use for DNS resolution.	A list of IP addresses as strings. For example, **["8.8.8.8", "192.168.1.12"]**.
platform: openstack: loadbalancer:	Whether or not to use the default, internal load balancer. If the value is set to **UserManaged**, this default load balancer is disabled so that you can deploy a cluster that uses an external, user-managed load balancer. If the parameter is not set, or if the value is **OpenShiftManagedDefault**, the cluster uses the default load balancer.	**UserManaged** or **OpenShiftManagedDefault**.
platform: openstack: machinesSubnet:	The UUID of a RHOSP subnet that the cluster's nodes use. Nodes and virtual IP (VIP) ports are created on this subnet. The first item in **networking.machineNetwork** must match the value of **machinesSubnet**. If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.	A UUID as a string. For example, **fa806b2f-ac49-4bce-b9db-124bc64209bf**.

11.1.7. Additional Google Cloud configuration parameters

Additional Google Cloud configuration parameters are described in the following table:

Table 11.7. Additional Google Cloud parameters

Parameter	Description	Values
controlPlane: platform: gcp: osImage: project:	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by specifying the location of a custom RHCOS image that the installation program is to use for control plane machines only. Control plane machines do not contribute to licensing costs when using the default image. But, if you apply a Google Cloud Marketplace image for a control plane machine, usage costs do apply.	String. The name of Google Cloud project where the image is located.
controlPlane: platform: gcp: osImage: name:	The name of the custom RHCOS image that the installation program is to use to boot control plane machines. If you use **controlPlane.platform.gcp.osImage.project**, this field is required.	String. The name of the RHCOS image.

Parameter	Description	Values
compute: platform: gcp: osImage: project:	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot compute machines. You can override the default behavior by specifying the location of a custom RHCOS image that the installation program is to use for compute machines only.	String. The name of Google Cloud project where the image is located.
compute: platform: gcp: osImage: name:	The name of the custom RHCOS image that the installation program is to use to boot compute machines. If you use **compute.platform.gcp.osImage.project**, this field is required.	String. The name of the RHCOS image.
platform: gcp: network:	The name of the existing Virtual Private Cloud (VPC) where you want to deploy your cluster. If you want to deploy your cluster into a shared VPC, you must set **platform.gcp.networkProjectID** with the name of the Google Cloud project that contains the shared VPC.	String.

Parameter	Description	Values
platform: gcp: networkProjectID:	Optional. The name of the Google Cloud project that contains the shared VPC where you want to deploy your cluster.	String.
platform: gcp: projectID:	The name of the Google Cloud project where the installation program installs the cluster.	String.
platform: gcp: region:	The name of the Google Cloud region that hosts your cluster.	Any valid region name, such as **us-central1**.

Parameter	Description	Values
platform: gcp: controlPlaneSubnet:	The name of the existing subnet where you want to deploy your control plane machines.	The subnet name.
platform: gcp: computeSubnet:	The name of the existing subnet where you want to deploy your compute machines.	The subnet name.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: zones:	The availability zones where the installation program creates machines.	A list of valid Google Cloud availability zones, such as **us-central1-a**, in a YAML sequence. **IMPORTANT** When running your cluster on Google Cloud 64-bit ARM infrastructures, ensure that you use a zone where Ampere Altra Arm CPU's are available. You can find which zones are compatible with 64-bit ARM processors in the "Google Cloud availability zones" link.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: diskSizeGB:	The size of the disk in gigabytes (GB).	Any size between 16 GB and 65536 GB.
platform: gcp: defaultMachinePlatform: osDisk: diskType:	The Google Cloud disk type.	The default disk type for all machines. Control plane nodes must use the **pd-ssd** disk type. Compute nodes can use the **pd-ssd**, **pd-balanced**, or **pd-standard** disk types.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osImage: project:	Optional. By default, the installation program downloads and installs the RHCOS image that is used to boot control plane and compute machines. You can override the default behavior by specifying the location of a custom RHCOS image that the installation program is to use for both types of machines.	String. The name of Google Cloud project where the image is located.
platform: gcp: defaultMachinePlatform: osImage: name:	The name of the custom RHCOS image that the installation program is to use to boot control plane and compute machines. If you use **platform.gcp.defaultMachinePlatform.osImage.project**, this field is required.	String. The name of the RHCOS image.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: tags:	Optional. Additional network tags to add to the control plane and compute machines.	One or more strings, for example **network-tag1**.
platform: gcp: defaultMachinePlatform: type:	The Google Cloud machine type for control plane and compute machines.	The Google Cloud machine type, for example **n1-standard-4**.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: name:	The name of the customer managed encryption key to be used for machine disk encryption.	The encryption key name.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: keyRing:	The name of the Key Management Service (KMS) key ring to which the KMS key belongs.	The KMS key ring name.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: location:	The Google Cloud location in which the KMS key ring exists.	The Google Cloud location.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: projectID:	The ID of the project in which the KMS key ring exists. This value defaults to the value of the **platform.gcp.projectID** parameter if it is not set.	The Google Cloud project ID.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKeyServiceAccount:	The Google Cloud service account used for the encryption request for control plane and compute machines. If absent, the Compute Engine default service account is used. For more information about Google Cloud service accounts, see Google's documentation on service accounts.	The Google Cloud service account email, for example **<service_account_name>@<project_id>.iam.gserviceaccount.com**.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: secureBoot:	Whether to enable Shielded VM secure boot for all machines in the cluster. Shielded VMs have additional security protocols such as secure boot, firmware and integrity monitoring, and rootkit protection. For more information on Shielded VMs, see Google's documentation on Shielded VMs.	**Enabled** or **Disabled**. The default value is **Disabled**.
platform: gcp: defaultMachinePlatform: confidentialCompute:	Whether to use Confidential VMs for all machines in the cluster. Confidential VMs provide encryption for data during processing. For more information on Confidential computing, see Google's documentation on Confidential computing.	**Enabled** or **Disabled**. The default value is **Disabled**.

Parameter	Description	Values
platform: gcp: defaultMachinePlatform: onHostMaintenance:	Specifies the behavior of all VMs during a host maintenance event, such as a software or hardware update. For Confidential VMs, this parameter must be set to **Terminate**. Confidential VMs do not support live VM migration.	**Terminate** or **Migrate**. The default value is **Migrate**.

Parameter	Description	Values
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: name:	The name of the customer managed encryption key to be used for control plane machine disk encryption.	The encryption key name.

Parameter	Description	Values
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	For control plane machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.

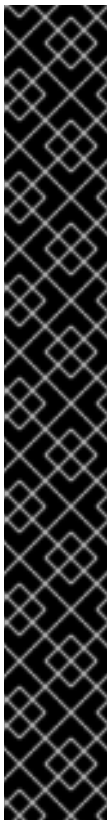
Parameter	Description	Values
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: location:	For control plane machines, the Google Cloud location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations.	The Google Cloud location for the key ring.

Parameter	Description	Values
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	For control plane machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The Google Cloud project ID.

Parameter	Description	Values
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	The Google Cloud service account used for the encryption request for control plane machines. If absent, the Compute Engine default service account is used. For more information about Google Cloud service accounts, see Google's documentation on service accounts.	The Google Cloud service account email, for example **<service_account_name>@<project_id>.iam.gserviceaccount.com**.

Parameter	Description	Values
controlPlane: platform: gcp: osDisk: diskSizeGB:	The size of the disk in gigabytes (GB). This value applies to control plane machines.	Any integer between 16 and 65536.
controlPlane: platform: gcp: osDisk: diskType:	The Google Cloud disk type for control plane machines.	Control plane machines must use the **pd-ssd** disk type, which is the default.

Parameter	Description	Values
controlPlane: platform: gcp: tags:	Optional. Additional network tags to add to the control plane machines. If set, this parameter overrides the **platform.gcp.defaultMachinePlatform.tags** parameter for control plane machines.	One or more strings, for example **control-plane-tag1**.
controlPlane: platform: gcp: type:	The Google Cloud machine type for control plane machines. If set, this parameter overrides the **platform.gcp.defaultMachinePlatform.type** parameter.	The Google Cloud machine type, for example **n1-standard-4**.

Parameter	Description	Values
controlPlane: platform: gcp: zones:	The availability zones where the installation program creates control plane machines.	A list of valid Google Cloud availability zones, such as **us-central1-a**, in a YAML sequence.  **IMPORTANT** When running your cluster on Google Cloud 64-bit ARM infrastructures, ensure that you use a zone where Ampere Altra Arm CPU's are available. You can find which zones are compatible with 64-bit ARM processors in the "Google Cloud availability zones" link.

Parameter	Description	Values
controlPlane: platform: gcp: secureBoot:	Whether to enable Shielded VM secure boot for control plane machines. Shielded VMs have additional security protocols such as secure boot, firmware and integrity monitoring, and rootkit protection. For more information on Shielded VMs, see Google's documentation on Shielded VMs.	**Enabled** or **Disabled**. The default value is **Disabled**.
controlPlane: platform: gcp: confidentialCompute:	Whether to enable Confidential VMs for control plane machines. Confidential VMs provide encryption for data while it is being processed. For more information on Confidential VMs, see Google's documentation on Confidential Computing.	**Enabled** or **Disabled**. The default value is **Disabled**.

Parameter	Description	Values
controlPlane: platform: gcp: onHostMaintenance:	Specifies the behavior of control plane VMs during a host maintenance event, such as a software or hardware update. For Confidential VMs, this parameter must be set to **Terminate**. Confidential VMs do not support live VM migration.	**Terminate** or **Migrate**. The default value is **Migrate**.
compute: platform: gcp: osDisk: encryptionKey: kmsKey: name:	The name of the customer managed encryption key to be used for compute machine disk encryption.	The encryption key name.

Parameter	Description	Values
compute: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	For compute machines, the name of the KMS key ring to which the KMS key belongs.	The KMS key ring name.

Parameter	Description	Values
compute: platform: gcp: osDisk: encryptionKey: kmsKey: location:	For compute machines, the Google Cloud location in which the key ring exists. For more information about KMS locations, see Google's documentation on Cloud KMS locations.	The Google Cloud location for the key ring.

Parameter	Description	Values
compute: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	For compute machines, the ID of the project in which the KMS key ring exists. This value defaults to the VM project ID if not set.	The Google Cloud project ID.

Parameter	Description	Values
compute: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	The Google Cloud service account used for the encryption request for compute machines. If this value is not set, the Compute Engine default service account is used. For more information about Google Cloud service accounts, see Google's documentation on service accounts.	The Google Cloud service account email, for example **<service_account_name>@<project_id>.iam.gserviceaccount.com**.
compute: platform: gcp: osDisk: diskSizeGB:	The size of the disk in gigabytes (GB). This value applies to compute machines.	Any integer between 16 and 65536.

Parameter	Description	Values
compute: platform: gcp: osDisk: diskType:	The Google Cloud disk type for compute machines.	**pd-ssd**, **pd-standard**, or **pd-balanced**. The default is **pd-ssd**.
compute: platform: gcp: tags:	Optional. Additional network tags to add to the compute machines. If set, this parameter overrides the **platform.gcp.defaultMachinePlatform.tags** parameter for compute machines.	One or more strings, for example **compute-network-tag1**.

Parameter	Description	Values
`compute:` `platform:` `gcp:` `type:`	The Google Cloud machine type for compute machines. If set, this parameter overrides the **platform.gcp.defaultMachinePlatform.type** parameter.	The Google Cloud machine type, for example **n1-standard-4**.
`compute:` `platform:` `gcp:` `zones:`	The availability zones where the installation program creates compute machines.	A list of valid Google Cloud availability zones, such as **us-central1-a**, in a YAML sequence. **IMPORTANT** When running your cluster on Google Cloud 64-bit ARM infrastructures, ensure that you use a zone where Ampere Altra Arm CPU's are available. You can find which zones are compatible with 64-bit ARM processors in the "Google Cloud availability zones" link.

Parameter	Description	Values
compute: platform: gcp: secureBoot:	Whether to enable Shielded VM secure boot for compute machines. Shielded VMs have additional security protocols such as secure boot, firmware and integrity monitoring, and rootkit protection. For more information on Shielded VMs, see Google's documentation on Shielded VMs.	**Enabled** or **Disabled**. The default value is **Disabled**.
compute: platform: gcp: confidentialCompute:	Whether to enable Confidential VMs for compute machines. Confidential VMs provide encryption for data while it is being processed. For more information on Confidential VMs, see Google's documentation on Confidential Computing.	**Enabled** or **Disabled**. The default value is **Disabled**.

Parameter	Description	Values
compute: platform: gcp: onHostMaintenance:	Specifies the behavior of compute VMs during a host maintenance event, such as a software or hardware update. For Confidential VMs, this parameter must be set to **Terminate**. Confidential VMs do not support live VM migration.	**Terminate** or **Migrate**. The default value is **Migrate**.