Red Hat OpenStack Platform 17.0

Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director

Configuring the director to deploy and use a Red Hat Ceph Storage cluster

Last Updated: 2023-09-06

Red Hat OpenStack Platform 17.0 Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director

Configuring the director to deploy and use a Red Hat Ceph Storage cluster

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

This guide provides information about using the Red Hat OpenStack Platform director to create an overcloud with a Red Hat Ceph Storage cluster. This includes instructions for customizing your Red Hat Ceph Storage cluster through the director.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Tell us how we can make it better.

Using the Direct Documentation Feedback (DDF) function

Use the **Add Feedback** DDF function for direct comments on specific sentences, paragraphs, or code blocks.

1. View the documentation in the *Multi-page HTML* format.
2. Ensure that you see the **Feedback** button in the upper right corner of the document.
3. Highlight the part of text that you want to comment on.
4. Click **Add Feedback**.
5. Complete the **Add Feedback** field with your comments.
6. Optional: Add your email address so that the documentation team can contact you for clarification on your issue.
7. Click **Submit**.

CHAPTER 1. DEPLOYING AN OVERCLOUD AND RED HAT CEPH STORAGE

Red Hat OpenStack Platform (RHOSP) director deploys the cloud environment, also known as the overcloud, and Red Hat Ceph Storage. Director uses Ansible playbooks provided through the **tripleo-ansible** package to deploy the Ceph Storage cluster. The director also manages the configuration and scaling operations of the Ceph Storage cluster.

For more information about Red Hat Ceph Storage, see Red Hat Ceph Storage Architecture Guide .

For more information about services in the Red Hat OpenStack Platform, see Configuring a basic overcloud with the CLI tools in *Director Installation and Usage*.

1.1. RED HAT CEPH STORAGE CLUSTERS

Red Hat Ceph Storage is a distributed data object store designed for performance, reliability, and scalability. Distributed object stores use unstructured data to simultaneously service modern and legacy object interfaces.

Ceph Storage is deployed as a cluster. A Ceph Storage cluster consists of two primary types of daemons:

- Ceph Object Storage Daemon (CephOSD) - The CephOSD performs data storage, data replication, rebalancing, recovery, monitoring, and reporting tasks.
- Ceph Monitor (CephMon) - The CephMon maintains the primary copy of the cluster map with the current state of the cluster.

For more information about Red Hat Ceph Storage, see the Red Hat Ceph Storage Architecture Guide .

1.2. RED HAT CEPH STORAGE NODE REQUIREMENTS

There are additional node requirements using director to create a Ceph Storage cluster:

- Hardware requirements including processor, memory, and network interface card selection and disk layout are available in the Red Hat Ceph Storage Hardware Guide .
- Each Ceph Storage node requires a supported power management interface, such as Intelligent Platform Management Interface (IPMI) functionality, on the motherboard of the server.
- Each Ceph Storage node must have at least two disks. RHOSP director uses **cephadm** to deploy the Ceph Storage cluster. The cephadm functionality does not support installing Ceph OSD on the root disk of the node.

1.3. CEPH STORAGE NODES AND RHEL COMPATIBILITY

RHOSP 17.0 is supported on RHEL 9.0. However, hosts that are mapped to the Ceph Storage role update to the latest major RHEL release. Before upgrading, review the Red Hat Knowledgebase article Red Hat Ceph Storage: Supported configurations .

1.4. DEPLOYING RED HAT CEPH STORAGE

You deploy Red Hat Ceph Storage in two phases:

- Create the Red Hat Ceph Storage cluster before deploying the overcloud.
- Configure the Red Hat Ceph Storage cluster during overcloud deployment.

A Ceph Storage cluster is created ready to serve the Ceph RADOS Block Device (RBD) service. Additionally, the following services are running on the appropriate nodes:

- Ceph Monitor (CephMon)
- Ceph Manager (CephMgr)
- Ceph OSD (CephOSD)

Pools and cephx keys are created during the configuration phase.

The following Ceph Storage components are not available until after the configuration phase:

- Ceph Dashboard (CephDashboard)
- Ceph Object Gateway (CephRGW)
- Ceph MDS (CephMds)

Red Hat Ceph Storage cluster configuration finalizes during overcloud deployment. Daemons and services such as Ceph Object Gateway and Ceph Dashboard deploy according to the overcloud definition. Red Hat OpenStack Platform (RHOSP) services are configured as Ceph Storage cluster clients.

1.5. RED HAT CEPH STORAGE DEPLOYMENT REQUIREMENTS

Provisioning of network resources and bare metal instances is required before Ceph Storage cluster creation. Configure the following before creating a Red Hat Ceph Storage cluster:

- Provision networks with the **openstack overcloud network provision** command and the **cli-overcloud-network-provision.yaml** ansible playbook.
- Provision bare metal instances with the **openstack overcloud node provision** command to provision bare metal instances using the **cli-overcloud-node-provision.yaml** ansible playbook.

For more information about these tasks, see:

- Networking Guide
- Bare Metal Provisioning

The following elements must be present in the overcloud environment to finalize the Ceph Storage cluster configuration:

- Red Hat OpenStack Platform director installed on an undercloud host. See Installing director in Director Installation and Usage.
- Installation of recommended hardware to support Red Hat Ceph Storage. For more information about recommended hardware, see the Red Hat Ceph Storage Hardware Guide.

1.6. POST DEPLOYMENT VERIFICATION

Director deploys a Ceph Storage cluster ready to serve Ceph RADOS Block Device (RBD) using **tripleo-ansible** roles executed by the **cephadm** command.

Verify the following are in place after **cephadm** completes Ceph Storage deployment:

- SSH access to a CephMon service node to use the **sudo cephadm shell** command.
- All OSDs operational.

NOTE

Check inoperative OSDs for environmental issues like uncleaned disks.

- A Ceph configuration file and client administration keyring file in the **/etc/ceph** directory of CephMon service nodes.
- The Ceph Storage cluster is ready to serve RBD.

Pools, cephx keys, CephDashboard, and CephRGW are configured during overcloud deployment by the **openstack overcloud deploy** command. This is for two reasons:

- The Dashboard and RGW services must integrate with **haproxy**. This is deployed with the overcloud.
- The creation of pools and cephx keys are dependent on which OpenStack clients are deployed.

These resources are created in the Ceph Storage cluster using the client administration keyring file and the **~/deployed_ceph.yaml** file output by the **openstack overcloud ceph deploy** command.

For more information about **cephadm**, see Red Hat Ceph Storage Installation Guide .

CHAPTER 2. PREPARING CEPH STORAGE NODES FOR DEPLOYMENT

Red Hat Ceph Storage nodes are bare metal systems with IPMI power management. Director installs Red Hat Enterprise Linux on each node.

Director communicates with each node through the Provisioning network during the introspection and provisioning processes. All nodes connect to the Provisioning network through the native VLAN.

For more information about bare metal provisioning before overcloud deployment, see Provisioning and deploying your overcloud in *Director Installation and Usage* guide.

For a complete guide to bare metal provisioning, see Bare Metal Provisioning.

2.1. CLEANING CEPH STORAGE NODE DISKS

Ceph Storage OSDs and journal partitions require factory clean disks. All data and metadata must be erased by the Bare Metal Provisioning service (ironic) from these disks before installing the Ceph OSD services.

You can configure director to delete all disk data and metadata by default by using the Bare Metal Provisioning service. When director is configured to perform this task, the Bare Metal Provisioning service performs an additional step to boot the nodes each time a node is set to **available**.

WARNING

The Bare Metal Provisioning service uses the **wipefs --force --all** command. This command deletes all data and metadata on the disk but it does not perform a secure erase. A secure erase takes much longer.

Procedure

1. Open **/home/stack/undercloud.conf** and add the following parameter:

```
clean_nodes=true
```

2. Save **/home/stack/undercloud.conf**.
3. Update the undercloud configuration.

```
openstack undercloud install
```

2.2. REGISTERING NODES

Register the nodes to enable communication with director.

Procedure

1. Create a node inventory JSON file in **/home/stack**.

2. Enter hardware and power management details for each node.
   For example:

```
{
    "nodes":[
        {
            "mac":[
                "b1:b1:b1:b1:b1:b1"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.205"
        },
        {
            "mac":[
                "b2:b2:b2:b2:b2:b2"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.206"
        },
        {
            "mac":[
                "b3:b3:b3:b3:b3:b3"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.207"
        },
        {
            "mac":[
                "c1:c1:c1:c1:c1:c1"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
```

```
        "pm_password":"p@55w0rd!",
        "pm_addr":"192.0.2.208"
    },
    {
        "mac":[
            "c2:c2:c2:c2:c2:c2"
        ],
        "cpu":"4",
        "memory":"6144",
        "disk":"40",
        "arch":"x86_64",
        "pm_type":"ipmi",
        "pm_user":"admin",
        "pm_password":"p@55w0rd!",
        "pm_addr":"192.0.2.209"
    },
    {
        "mac":[
            "c3:c3:c3:c3:c3:c3"
        ],
        "cpu":"4",
        "memory":"6144",
        "disk":"40",
        "arch":"x86_64",
        "pm_type":"ipmi",
        "pm_user":"admin",
        "pm_password":"p@55w0rd!",
        "pm_addr":"192.0.2.210"
    },
    {
        "mac":[
            "d1:d1:d1:d1:d1:d1"
        ],
        "cpu":"4",
        "memory":"6144",
        "disk":"40",
        "arch":"x86_64",
        "pm_type":"ipmi",
        "pm_user":"admin",
        "pm_password":"p@55w0rd!",
        "pm_addr":"192.0.2.211"
    },
    {
        "mac":[
            "d2:d2:d2:d2:d2:d2"
        ],
        "cpu":"4",
        "memory":"6144",
        "disk":"40",
        "arch":"x86_64",
        "pm_type":"ipmi",
        "pm_user":"admin",
        "pm_password":"p@55w0rd!",
        "pm_addr":"192.0.2.212"
    },
    {
```

```
            "mac":[
                "d3:d3:d3:d3:d3:d3"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.213"
        }
    ]
}
```

3. Save the new file.
4. Initialize the stack user:

```
$ source ~/stackrc
```

5. Import the JSON inventory file into director and register nodes

```
$ openstack overcloud node import <inventory_file>
```

Replace **<inventory_file>** with the name of the file created in the first step.

6. Assign the kernel and ramdisk images to each node:

```
$ openstack overcloud node configure <node>
```

2.3. VERIFYING AVAILABLE RED HAT CEPH STORAGE PACKAGES

Verify all required packages are available to avoid overcloud deployment failures.

2.3.1. Verifying cephadm package installation

Verify the **cephadm** package is installed on at least one overcloud node. The **cephadm** package is used to bootstrap the first node of the Ceph Storage cluster.

The **cephadm** package is included in the **overcloud-hardened-uefi-full.qcow2** image. The **tripleo_cephadm** role uses the Ansible package module to ensure it is present in the image.

2.4. DEFINING THE ROOT DISK FOR MULTI-DISK CEPH CLUSTERS

Ceph Storage nodes typically use multiple disks. Director must identify the root disk in multiple disk configurations. The overcloud image is written to the root disk during the provisioning process.

Hardware properties are used to identify the root disk. For more information about properties you can use to identify the root disk, see Section 2.4.1, “Properties that identify the root disk” .

Procedure

1. Verify the disk information from the hardware introspection of each node:

```
(undercloud)$ openstack baremetal introspection data save <node_uuid> | --file
<output_file_name>
```

   - Replace **<node_uuid>** with the UUID of the node.
   - Replace **<output_file_name>** with the name of the file that contains the output of the node introspection.
     For example, the data for one node might show three disks:


```
[
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sda",
    "wwn_vendor_extension": "0x1ea4dcc412a9632b",
    "wwn_with_extension": "0x61866da04f3807001ea4dcc412a9632b",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f380700",
    "serial": "61866da04f3807001ea4dcc412a9632b"
  }
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sdb",
    "wwn_vendor_extension": "0x1ea4e13c12e36ad6",
    "wwn_with_extension": "0x61866da04f380d001ea4e13c12e36ad6",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f380d00",
    "serial": "61866da04f380d001ea4e13c12e36ad6"
  }
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sdc",
    "wwn_vendor_extension": "0x1ea4e31e121cfb45",
    "wwn_with_extension": "0x61866da04f37fc001ea4e31e121cfb45",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f37fc00",
    "serial": "61866da04f37fc001ea4e31e121cfb45"
  }
]
```


2. Set the root disk for the node by using a unique hardware property:
   (undercloud)$ openstack baremetal node set --property root_device='{<property_value>}' <node-uuid>

   - Replace **<property_value>** with the unique hardware property value from the introspection data to use to set the root disk.
   - Replace **<node_uuid>** with the UUID of the node.

NOTE

A unique hardware property is any property from the hardware introspection step that uniquely identifies the disk. For example, the following command uses the disk serial number to set the root disk:

```
(undercloud)$ openstack baremetal node set --property
root_device='{"serial": "61866da04f380d001ea4e13c12e36ad6"}'
1a4e30da-b6dc-499d-ba87-0bd8a3819bc0
```

3. Configure the BIOS of each node to first boot from the network and then the root disk.

Director identifies the specific disk to use as the root disk. When you run the **openstack overcloud node provision** command, director provisions and writes the overcloud image to the root disk.

2.4.1. Properties that identify the root disk

There are several properties that you can define to help director identify the root disk:

- **model** (String): Device identifier.
- **vendor** (String): Device vendor.
- **serial** (String): Disk serial number.
- **hctl** (String): Host:Channel:Target:Lun for SCSI.
- **size** (Integer): Size of the device in GB.
- **wwn** (String): Unique storage identifier.
- **wwn_with_extension** (String): Unique storage identifier with the vendor extension appended.
- **wwn_vendor_extension** (String): Unique vendor storage identifier.
- **rotational** (Boolean): True for a rotational device (HDD), otherwise false (SSD).
- **name** (String): The name of the device, for example: /dev/sdb1.

IMPORTANT

Use the **name** property for devices with persistent names. Do not use the **name** property to set the root disk for devices that do not have persistent names because the value can change when the node boots.

2.5. USING THE OVERCLOUD-MINIMAL IMAGE TO AVOID USING A RED HAT SUBSCRIPTION ENTITLEMENT

The default image for a Red Hat OpenStack Platform (RHOSP) deployment is **overcloud-hardened-uefi-full.qcow2**. The **overcloud-hardened-uefi-full.qcow2** image uses a valid Red Hat OpenStack Platform (RHOSP) subscription. You can use the **overcloud-minimal** image when you do not want to consume your subscription entitlements, to avoid reaching the limit of your paid Red Hat subscriptions. This is useful, for example, when you want to provision nodes with only Ceph daemons, or when you want

to provision a bare operating system (OS) where you do not want to run any other OpenStack services. For information about how to obtain the **overcloud-minimal** image, see Obtaining images for overcloud nodes.

NOTE

The **overcloud-minimal** image supports only standard Linux bridges. The **overcloud-minimal** image does not support Open vSwitch (OVS) because OVS is an OpenStack service that requires a Red Hat OpenStack Platform subscription entitlement. OVS is not required to deploy Ceph Storage nodes. Use **linux_bond** instead of **ovs_bond** to define bonds.

Procedure

1. Open your **/home/stack/templates/overcloud-baremetal-deploy.yaml** file.
2. Add or update the **image** property for the nodes that you want to use the **overcloud-minimal** image. You can set the image to **overcloud-minimal** on specific nodes, or for all nodes for a role.

NOTE

The overcloud minimal image is not a whole disk image. The kernel and ramdisk must be specified in the **/home/stack/templates/overcloud-baremetal-deploy.yaml** file.

Specific nodes

```
- name: Ceph
  count: 3
  instances:
  - hostname: overcloud-ceph-0
    name: node00
    image:
      href: file:///var/lib/ironic/images/overcloud-minimal.raw
      kernel: file://var/lib/ironic/images/overcloud-minimal.vmlinuz
      ramdisk: file://var/lib/ironic/images/overcloud-minimal.initrd
  - hostname: overcloud-ceph-1
    name: node01
    image:
      href: file:///var/lib/ironic/images/overcloud-minimal.raw
      kernel: file://var/lib/ironic/images/overcloud-minimal.vmlinuz
      ramdisk: file://var/lib/ironic/images/overcloud-minimal.initrd
  - hostname: overcloud-ceph-2
    name: node02
    image:
      href: file:///var/lib/ironic/images/overcloud-minimal.raw
      kernel: file://var/lib/ironic/images/overcloud-minimal.vmlinuz
      ramdisk: file://var/lib/ironic/images/overcloud-minimal.initrd
```

All nodes for a specific role

```
- name: Ceph
  count: 3
```

```
defaults:
  image:
    href: file:///var/lib/ironic/images/overcloud-minimal.raw
    kernel: file://var/lib/ironic/images/overcloud-minimal.vmlinuz
    ramdisk: file://var/lib/ironic/images/overcloud-minimal.initrd
instances:
- hostname: overcloud-ceph-0
  name: node00
- hostname: overcloud-ceph-1
  name: node01
- hostname: overcloud-ceph-2
  name: node02
```

3. In the **roles_data.yaml** role definition file, set the **rhsm_enforce** parameter to **False**.

```
rhsm_enforce: False
```

4. Run the provisioning command:

```
(undercloud)$ openstack overcloud node provision \
--stack overcloud \
--output /home/stack/templates/overcloud-baremetal-deployed.yaml \
/home/stack/templates/overcloud-baremetal-deploy.yaml
```

5. Pass the **overcloud-baremetal-deployed.yaml** environment file to the **openstack overcloud ceph deploy** command.

2.6. DESIGNATING NODES FOR RED HAT CEPH STORAGE

To designate nodes for Red Hat Ceph Storage, you must create a new role file to configure the **CephStorage** role, and configure the bare metal nodes with a resource class for **CephStorage**.

Procedure

1. Log in to the undercloud as the **stack** user.
2. Source the **stackrc** file:

```
[stack@director ~]$ source ~/stackrc
```

3. Generate a new roles data file named **roles_data.yaml** that includes the **Controller**, **Compute**, and **CephStorage** roles:

```
(undercloud)$ openstack overcloud roles \
 generate Controller Compute CephStorage -o /home/stack/templates/roles_data.yaml \
```

4. Open **roles_data.yaml** and ensure it has the following parameters and sections:

Section/Parameter	Value
Role comment	**Role: CephStorage**

Section/Parameter	Value
Role name	**name: CephStorage**
description	**Ceph node role**
HostnameFormatDefault	**%stackname%-novaceph-%index%**
deprecated_nic_config_name	**ceph.yaml**

5. Register the Ceph nodes for the overcloud by adding them to your node definition template.
6. Inspect the node hardware:

```
(undercloud)$ openstack overcloud node introspect --all-manageable --provide
```

7. Tag each bare metal node that you want to designate for Ceph with a custom Ceph resource class:

```
(undercloud)$ openstack baremetal node set \
 --resource-class baremetal.CEPH <node>
```

Replace **<node>** with the ID of the bare metal node.

8. Add the **CephStorage** role to your **overcloud-baremetal-deploy.yaml** file, and define any predictive node placements, resource classes, or other attributes that you want to assign to your nodes:

```
- name: Controller
    count: 3
- name: Compute
   count: 3
- name: CephStorage
  count: 5
   defaults:
    resource_class: baremetal.CEPH
```

9. Run the provisioning command:

```
(undercloud)$ openstack overcloud node provision \
--stack stack \
--output /home/stack/templates/overcloud-baremetal-deployed.yaml \
/home/stack/templates/overcloud-baremetal-deploy.yaml
```

10. Monitor the provisioning progress in a separate terminal. When provisioning is successful, the node state changes from **available** to **active**:

```
(undercloud)$ watch openstack baremetal node list
```

Additional resources

- For more information on node registration, see Section 2.2, "Registering nodes".
- For more information inspecting node hardware, see Creating an inventory of the bare-metal node hardware in the *Director Installation and Usage* guide.

CHAPTER 3. CONFIGURING THE RED HAT CEPH STORAGE CLUSTER

To deploy the Red Hat Ceph Storage cluster for your Red Hat OpenStack Platform environment, you must first configure the Red Hat Ceph Storage cluster options for your environment.

Configure the Red Hat Ceph Storage cluster options:

- Configuring time synchronization
- Configuring the Red Hat Ceph Storage cluster name
- Configuring network options with the network data file
- Configuring network options with a configuration file
- Configuring a CRUSH hierarchy for an OSD
- Configuring Ceph service placement options
- Configuring SSH user options for Ceph nodes
- Configuring the container registry

Prerequisites

Before you can configure and deploy the Red Hat Ceph Storage cluster, use the the Bare Metal Provisioning service (ironic) to provision the bare metal instances and networks. For more information, see Bare Metal Provisioning.

3.1. THE OPENSTACK OVERCLOUD CEPH DEPLOY COMMAND

If you deploy the Ceph cluster using director, you must use the **openstack overcloud ceph deploy** command. For a complete listing of command options and parameters, see openstack overcloud ceph deploy in the *Command Line Interface Reference*.

The command **openstack overcloud ceph deploy --help** provides the current options and parameters available in your environment.

3.2. CEPH CONFIGURATION FILE

A standard format initialization file is one way to perform Ceph cluster configuration. This initialization file is used to configure the Ceph cluster. Use one of the following commands to use this file: * **cephadm bootstap --config <file_name>** * **openstack overcloud ceph deploy --config <file_name>** commands.

Example

The following example creates a simple initialization file called **initial-ceph.conf** and then uses the **openstack overcloud ceph deploy** command to configure the Ceph cluster with it. It demonstrates how to configure the messenger v2 protocol to use a secure mode that encrypts all data passing over the network.

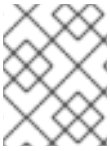
```
$ cat <<EOF > initial-ceph.conf
[global]
```

```
ms_cluster_mode: secure
ms_service_mode: secure
ms_client_mode: secure
EOF
$ openstack overcloud ceph deploy --config initial-ceph.conf ...
```

3.3. CONFIGURING TIME SYNCHRONIZATION

The Time Synchronization Service (chrony) is enabled for time synchronization by default. You can perform the following tasks to configure the service.

- Configuring time synchronization with a delimited list
- Configuring time synchronization with an environment file
- Disabling time synchronization



NOTE

Time synchronization is configured using either a delimited list or an environment file. Use the procedure that is best suited to your administrative practices.

3.3.1. Configuring time synchronization with a delimited list

You can configure the Time Synchronization Service (chrony) to use a delimited list to configure NTP servers.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Configure NTP servers with a delimited list:

```
openstack overcloud ceph deploy \
      --ntp-server "<ntp_server_list>"
```

Replace **<ntp_server_list>** with a comma delimited list of servers.

```
openstack overcloud ceph deploy \
      --ntp-server "0.pool.ntp.org,1.pool.ntp.org"
```

3.3.2. Configuring time synchronization with an environment file

You can configure the Time Synchronization Service (chrony) to use an environment file that defines NTP servers.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create an environment file, such as **/home/stack/templates/ntp-parameters.yaml**, to contain the NTP server configuration.

3. Add the **NtpServer** parameter. The **NtpServer** parameter contains a comma delimited list of NTP servers.

```
parameter_defaults:
  NtpServer: 0.pool.ntp.org,1.pool.ntp.org
```

4. Configure NTP servers with an environment file:

```
openstack overcloud ceph deploy \
      --ntp-heat-env-file "<ntp_file_name>"
```

Replace **<ntp_file_name>** with the name of the environment file you created.

```
openstack overcloud ceph deploy \
      --ntp-heat-env-file "/home/stack/templates/ntp-parameters.yaml"
```

3.3.3. Disabling time synchronization

The Time Synchronization Service (chrony) is enabled by default. You can disable the service if you do not want to use it.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Disable the Time Synchronization Service (chrony):

```
openstack overcloud ceph deploy \
      --skip-ntp
```

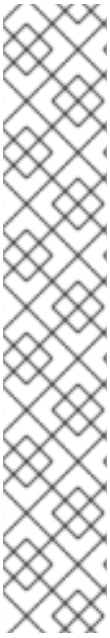
3.4. CONFIGURING THE RED HAT CEPH STORAGE CLUSTER NAME

You can deploy the Red Hat Ceph Storage cluster with a name that you configure. The default name is **ceph**.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Configure the name of the Ceph Storage cluster by using the following command: **openstack overcloud ceph deploy** \ **--cluster <cluster_name>**

   $ openstack overcloud ceph deploy \ **--cluster central** \



NOTE

Keyring files are not created at this time. Keyring files are created during the overcloud deployment. Keyring files inherit the cluster name configured during this procedure. For more information about overcloud deployment see Section 8.1, "Initiating overcloud deployment"

In the example above, the Ceph cluster is named **central**. The configuration and keyring files for the **central** Ceph cluster would be created in **/etc/ceph** during the deployment process.

```
[root@oc0-controller-0 ~]# ls -l /etc/ceph/
total 16
-rw-------. 1 root root  63 Mar 26 21:49 central.client.admin.keyring
-rw-------. 1  167  167 201 Mar 26 22:17 central.client.openstack.keyring
-rw-------. 1  167  167 134 Mar 26 22:17 central.client.radosgw.keyring
-rw-r--r--. 1 root root 177 Mar 26 21:49 central.conf
```

Troubleshooting

The following error may be displayed if you configure a custom name for the Ceph Storage cluster:

monclient: get_monmap_and_config cannot identify monitors to contact because

If this error is displayed, use the following command after Ceph deployment:

cephadm shell --config <configuration_file> --keyring <keyring_file>

For example, if this error was displayed when you configured the cluster name to **central**, you would use the following command:

```
cephadm shell --config /etc/ceph/central.conf \
              --keyring /etc/ceph/central.client.admin.keyring
```

The following command could also be used as an alternative:

```
cephadm shell --mount /etc/ceph:/etc/ceph
export CEPH_ARGS='--cluster central'
```

3.5. CONFIGURING NETWORK OPTIONS WITH THE NETWORK DATA FILE

The network data file describes the networks used by the Red Hat Ceph Storage cluster.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create a YAML format file that defines the custom network attributes called **network_data.yaml**.

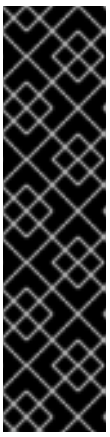
IMPORTANT

Using network isolation, the standard network deployment consists of two storage networks which map to the two Ceph networks:

- The storage network, **storage**, maps to the Ceph network, **public_network**. This network handles storage traffic such as the RBD traffic from the Compute nodes to the Ceph cluster.
- The storage network, **storage_mgmt**, maps to the Ceph network, **cluster_network**. This network handles storage management traffic such as data replication between Ceph OSDs.

3. Use the **openstack overcloud ceph deploy** command with the **--crush-hierarchy** option to deploy the configuration.

```
openstack overcloud ceph deploy \
      deployed_metal.yaml \
      -o deployed_ceph.yaml \
      --network-data network_data.yaml
```



IMPORTANT

The **openstack overcloud ceph deploy** command uses the network data file specified by the **--network-data** option to determine the networks to be used as the **public_network** and **cluster_network**. The command assumes these networks are named **storage** and **storage_mgmt** in network data file unless a different name is specified by the **--public-network-name** and **--cluster-network-name** options.

You must use the **--network-data** option when deploying with network isolation. The default undercloud (192.168.24.0/24) will be used for both the **public_network** and **cluster_network** if you do not use this option.

3.6. CONFIGURING NETWORK OPTIONS WITH A CONFIGURATION FILE

Network options can be specified with a configuration file as an alternative to the network data file.

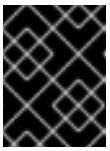
IMPORTANT

Using this method to configure network options overwrites automatically generated values in **network_data.yaml**. Ensure you set all four values when using this network configuration method.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create a standard format initialization file to configure the Ceph cluster. If you have already created a file to include other configuration options, you can add the network configuration to it.
3. Add the following parameters to the **[global]** section of the file:
   - **public_network**

- **public_network**
- **cluster_network**
- **ms_bind_ipv4**



IMPORTANT

Ensure the **public_network** and **cluster_network** map to the same networks as **storage** and **storage_mgmt**.

The following is an example of a configuration file entry for a network configuration with multiple subnets and custom networking names:

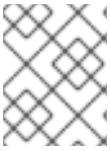
```
[global]
public_network = 172.16.14.0/24,172.16.15.0/24
cluster_network = 172.16.12.0/24,172.16.13.0/24
ms_bind_ipv4 = True
ms_bind_ipv6 = False
```

4. Use the command **openstack overcloud ceph deploy** with the **--config** option to deploy the configuration file.

```
$ openstack overcloud ceph deploy \
 --config initial-ceph.conf --network-data network_data.yaml
```

3.7. CONFIGURING A CRUSH HIERARCHY FOR AN OSD

You can configure a custom Controlled Replication Under Scalable Hashing (CRUSH) hierarchy during OSD deployment to add the OSD **location** attribute to the Ceph Storage cluster **hosts** specification. The **location** attribute configures where the OSD is placed within the CRUSH hierarchy.



NOTE

The **location** attribute sets only the initial CRUSH location. Subsequent changes of the attribute are ignored.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Source the **stackrc** undercloud credentials file:
   $ source ~/stackrc
3. Create a configuration file to define the custom CRUSH hierarchy, for example, **crush_hierarchy.yaml**.
4. Add the following configuration to the file:

```
ceph_crush_hierarchy:
 <osd_host>:
  root: default
  rack: <rack_num>
 <osd_host>:
```

```
    root: default
    rack: <rack_num>
  <osd_host>:
    root: default
    rack: <rack_num>
```

- Replace **<osd_host>** with the hostnames of the nodes where the OSDs are deployed, for example, **ceph-0**.
- Replace **<rack_num>** with the number of the rack where the OSDs are deployed, for example, **r0**.

5. Deploy the Ceph cluster with your custom OSD layout:

```
openstack overcloud ceph deploy \
      deployed_metal.yaml \
      -o deployed_ceph.yaml \
      --osd-spec osd_spec.yaml \
      --crush-hierarchy crush_hierarchy.yaml
```

The Ceph cluster is created with the custom OSD layout.

The example file above would result in the following OSD layout.

```
ID  CLASS  WEIGHT      TYPE NAME              STATUS  REWEIGHT  PRI-AFF
-1         0.02939     root default
-3         0.00980     rack r0
-2         0.00980         host ceph-node-00
 0   hdd   0.00980             osd.0              up   1.00000   1.00000
-5         0.00980     rack r1
-4         0.00980         host ceph-node-01
 1   hdd   0.00980             osd.1              up   1.00000   1.00000
-7         0.00980     rack r2
-6         0.00980         host ceph-node-02
 2   hdd   0.00980             osd.2              up   1.00000   1.00000
```

NOTE

Device classes are automatically detected by Ceph but CRUSH rules are associated with pools. Pools are still defined and created using the **CephCrushRules** parameter during the overcloud deployment.

Additional resources

See Red Hat Ceph Storage workload considerations in the *Red Hat Ceph Storage Installation Guide* for additional information.

3.8. CONFIGURING CEPH SERVICE PLACEMENT OPTIONS

You can define what nodes run what Ceph services using a custom roles file. A custom roles file is only necessary when default role assignments are not used because of the environment. For example, when deploying hyperconverged nodes, the predeployed compute nodes should be labeled as **osd** with a service type of **osd** to have a placement list containing a list of compute instances.

Service definitions in the **roles_data.yaml** file determine which bare metal instance runs which service. By default, the Controller role has the CephMon and CephMgr service while the CephStorage role has the CephOSD service. Unlike most composable services, Ceph services do not require heat output to determine how services are configured. The **roles_data.yaml** file always determines Ceph service placement even though the deployed Ceph process occurs before Heat runs.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create a YAML format file that defines the custom roles.
3. Deploy the configuration file:

```
$ openstack overcloud ceph deploy \
      deployed_metal.yaml \
      -o deployed_ceph.yaml \
      --roles-data custom_roles.yaml
```

3.9. CONFIGURING SSH USER OPTIONS FOR CEPH NODES

The **openstack overcloud ceph deploy** command creates the user and keys and distributes them to the hosts so it is not necessary to perform the procedures in this section. However, it is a supported option.

Cephadm connects to all managed remote Ceph nodes using SSH. The Red Hat Ceph Storage cluster deployment process creates an account and SSH key pair on all overcloud Ceph nodes. The key pair is then given to Cephadm so it can communicate with the nodes.

3.9.1. Creating the SSH user before Red Hat Ceph Storage cluster creation

You can create the SSH user before Ceph cluster creation with the **openstack overcloud ceph user enable** command.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create the SSH user:
   $ openstack overcloud ceph user enable

NOTE

The default user name is **ceph-admin**. To specify a different user name, use the **--cephadm-ssh-user** option to specify a different one.

openstack overcloud ceph user enable --cephadm-ssh-user <custom_user_name>

It is recommended to use the default name and not use the **--cephadm-ssh-user** parameter.

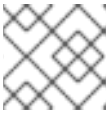
If the user is created in advance, use the parameter **--skip-user-create** when executing **openstack overcloud ceph deploy**.

3.9.2. Disabling the SSH user

Disabling the SSH user disables Cephadm. Disabling Cephadm removes the ability of the service to administer the Ceph cluster and prevents associated commands from working. It also prevents Ceph node overcloud scaling operations. It also removes all public and private SSH keys.

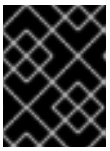
Procedure

1. Log in to the undercloud node as the **stack** user.
2. Use the command **openstack overcloud ceph user disable --fsid <FSID> ceph_spec.yaml** to disable the SSH user.



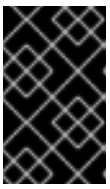
NOTE

The FSID is located in the **deployed_ceph.yaml** environment file.



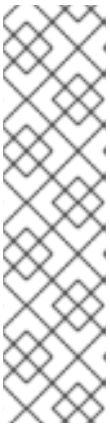
IMPORTANT

The **openstack overcloud ceph user disable** command is not recommended unless it is necessary to disable Cephadm.



IMPORTANT

To enable the SSH user and Cephadm service after being disabled, use the **openstack overcloud ceph user enable --fsid <FSID> ceph_spec.yaml** command.



NOTE

This command requires the path to a Ceph specification file to determine:

- Which hosts require the SSH user.
- Which hosts have the **_admin** label and require the private SSH key.
- Which hosts require the public SSH key.

For more information about specification files and how to generate them, see Generating the service specification.

3.10. ACCESSING CEPH STORAGE CONTAINERS

Obtaining and modifying container images in the *Transitioning to Containerized Services* guide contains procedures and information on how to prepare the registry and your undercloud and overcloud configuration to use container images. Use the information in this section to adapt these procedures to access Ceph Storage containers.

There are two options for accessing Ceph Storage containers from the overcloud.

- Downloading containers directly from a remote registry
- Cacheing containers on the undercloud

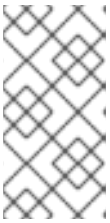
3.10.1. Downloading containers directly from a remote registry

You can configure Ceph to download containers directly from a remote registry.

Procedure

1. Create a **containers-prepare-parameter.yaml** file using the procedure Preparing container images.
2. Add the remote registry credentials to the **containers-prepare-parameter.yaml** file using the **ContainerImageRegistryCredentials** parameter as described in Obtaining container images from private registries.
3. When you deploy Ceph, pass the **containers-prepare-parameter.yaml** file using the **openstack overcloud ceph deploy** command.

```
openstack overcloud ceph deploy \
      --container-image-prepare containers-prepare-parameter.yaml
```



NOTE

If you do not cache the containers on the undercloud, as described in Cacheing containers on the undercloud, then you should pass the same **containers-prepare-parameter.yaml** file to the **openstack overcloud ceph deploy** command when you deploy Ceph. This will cache containers on the undercloud.

Result

The credentials in the **containers-prepare-parameter.yaml** are used by the **cephadm** command to authenticate to the remote registry and download the Ceph Storage container.

3.10.2. Cacheing containers on the undercloud

The procedure Modifying images during preparation describes using the following command:

```
sudo openstack tripleo container image prepare \
  -e ~/containers-prepare-parameter.yaml \
```

If you do not use the **--container-image-prepare** option to provide authentication credentials to the **openstack overcloud ceph deploy** command and directly download the Ceph containers from a remote registry, as described in Downloading containers directly from a remote registry , you must run the **sudo openstack tripleo container image prepare** command before deploying Ceph.

CHAPTER 4. CUSTOMIZING THE RED HAT CEPH STORAGE CLUSTER

Director deploys Red Hat Ceph Storage with a default configuration. You can customize this default configuration.

Prerequisites

- Ceph Storage nodes deployed with their storage network configured.
- The deployed bare metal file output by **openstack overcloud node provision -o ~/deployed_metal.yaml …**.

4.1. CONFIGURATION OPTIONS

There are several options for configuring the Red Hat Ceph Storage cluster.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Optional: Use a standard format initialization (ini) file to configure the Ceph cluster.
   a. Create the file with configuration options.
   The following is an example of a simple configuration file:

```
[global]
osd crush chooseleaf type = 0
log_file = /var/log/ceph/$cluster-$type.$id.log

[mon]
mon_cluster_log_to_syslog = true
```

   b. Save the configuration file.
   c. Use the **openstack overcloud ceph deploy --config <configuration_file_name>** command to deploy the configuration.
   Replace **<configuration_file_name>** with the name of the file you created.

```
$ openstack overcloud ceph deploy --config initial-ceph.conf
```

3. Optional: Send configuration values to the **cephadm bootstrap** command:
   openstack overcloud ceph deploy --force \ **--cephadm-extra-args '<optional_arguments>'** \

   Replace **<optional_arguments>** with the configuration values to provide to the underlying command.

NOTE

When using the arguments **--log-to-file** and **--skip-prepare-host**, the command **openstack overcloud ceph deploy --force** \ **--cephadm-extra-args '--log-to-file --skip-prepare-host'** \ is used.

4.2. GENERATING THE SERVICE SPECIFICATION (OPTIONAL)

The Red Hat Ceph Storage cluster service specification is a YAML file that describes the deployment of Ceph Storage services. It is automatically generated by **tripleo** before the Ceph Storage cluster is deployed. It does not usually have to be generated separately.

A custom service specification can be created to customize the Red Hat Ceph Storage cluster.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Generate the specification file:
   openstack overcloud ceph spec -o '<specification_file>'

   Replace **<specification_file>** with the name of the file to generate with the current service specification.

   openstack overcloud ceph spec -o '~/ceph_spec.yaml'
3. Edit the generated file with the required configuration.
4. Deploy the custom service specification:
   openstack overcloud ceph deploy \ deployed_metal.yaml \ -o deployed_ceph.yaml \ --ceph-spec <specification_file>

   Replace **<specification_file>** with the name of the custom service specification file.

   openstack overcloud ceph deploy \ deployed_metal.yaml \ -o deployed_ceph.yaml \ --ceph-spec ~/ceph_spec.yaml

4.3. CEPH CONTAINERS FOR RED HAT OPENSTACK PLATFORM WITH RED HAT CEPH STORAGE

You must have a Ceph Storage container to configure Red Hat Openstack Platform (RHOSP) to use Red Hat Ceph Storage with NFS Ganesha. You do not require a Ceph Storage container if the external Ceph Storage cluster only provides Block (through RBD), Object (through RGW), or File (through native CephFS) storage.

RHOSP 17.0 requires Red Hat Ceph Storage 5.x (Ceph package 16.x) or later to be compatible with Red Hat Enterprise Linux 9. The Ceph Storage 5.x containers are hosted at **registry.redhat.io**, a registry that requires authentication. For more information, see Container image preparation parameters.

4.4. CONFIGURING ADVANCED OSD SPECIFICATIONS

Configure an advanced OSD specification when the default specification does not provide the necessary functionality for your Ceph Storage cluster.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create a YAML format file that defines the advanced OSD specification.
   The following is an example of a custom OSD specification.

```
data_devices:
  rotational: 1
db_devices:
  rotational: 0
```

This example would create an OSD specification where all rotating devices will be data devices and all non-rotating devices will be used as shared devices. When the dynamic Ceph service specification is built, whatever is in the specification file is appended to the section of the specification if the **service_type** is **osd**.

3. Save the specification file.

4. Deploy the specification:
   openstack overcloud ceph deploy \ --osd-spec <osd_specification_file>

   Replace **<osd_specification_file>** with the name of the specification file you created.

   **$ openstack overcloud ceph deploy \ --osd-spec osd_spec.yaml **

Additional resources

For a list of OSD-related attributes used to configure OSDs in the service specification, see Advanced service specifications and filters for deploying OSDs in the *Red Hat Ceph Storage Operations Guide* .

4.5. MIGRATING FROM NODE-SPECIFIC OVERRIDES

Node-specific overrides were used to manage non-homogenous server hardware before Red Hat OpenStack Platform 17.0. This is now done with a custom OSD specification file. See Configuring advanced OSD specifications for information on how to create a custom OSD specification file.

4.6. ENABLING CEPH ON-WIRE ENCRYPTION

Enable encryption for all Ceph Storage traffic using the **secure mode** of the messenger version 2 protocol. Configure Ceph Storage as described in Encryption and Key Management in the Red Hat Ceph Storage *Data Security and Hardening Guide* to enable Ceph on-wire encryption.

Additional resources

For more information about Ceph on-wire encryption, see Ceph on-wire encryption in the Red Hat Ceph Storage *Architecture Guide*.

CHAPTER 5. CUSTOMIZING THE STORAGE SERVICE

The director heat template collection contains the necessary templates and environment files to enable a basic Ceph Storage configuration.

Director uses the **/usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml** environment file to add onfiguration to the Ceph Storage cluster deployed by **openstack overcloud ceph deploy** and integrate it with your overcloud during deployment.

5.1. CONFIGURING A CUSTOM ENVIRONMENT FILE

Director applies basic, default settings to the deployed Red Hat Ceph Storage cluster. You must define additional configuration in a custom environment file.

Procedure

1. Log in to the undercloud as the **stack** user.
2. Create a file to define the custom configuration.
   vi /home/stack/templates/storage-config.yaml
3. Add a **parameter_defaults** section to the file.
4. Add the custom configuration parameters. For more information about parameter definitions, see Overcloud Parameters.

```
parameter_defaults:
  CinderEnableIscsiBackend: false
  CinderEnableRbdBackend: true
  CinderBackupBackend: ceph
  NovaEnableRbdBackend: true
  GlanceBackend: rbd
```

NOTE

Parameters defined in a custom configuration file override any corresponding default settings in **/usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml**.

5. Save the file.

Additional resources

The custom configuration is applied during overcloud deployment.

5.2. RED HAT CEPH STORAGE PLACEMENT GROUPS

Placement groups (PGs) facilitate dynamic and efficient object tracking at scale. In the event of OSD failure or Ceph Storage cluster rebalancing, Ceph can move or replicate a placement group and the contents of the placement group. This allows a Ceph Storage cluster to rebalance and recover efficiently.

The placement group and replica count settings are not changed from the defaults unless the following parameters are included in a Ceph configuration file:

- **osd_pool_default_size**
- **osd_pool_default_pg_num**
- **osd_pool_default_pgp_num**

When the overcloud is deployed with the **openstack overcloud deploy** command, a pool is created for every enabled Red Hat OpenStack Platform service. For example, the following command creates pools for the Compute service (nova), the Block Storage service (cinder), and the Image service (glance):

```
openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm-rbd-only.yaml
```

Adding **-e environments/cinder-backup.yaml** to the command, creates a pool called **backups**:

```
openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm-rbd-only.yaml
  -e environments/cinder-backup.yaml
```

It is not necessary to configure a placement group number per pool; the **pg_autoscale_mode** attribute is enabled by default. However, it is recommended to configure the **target_size_ratio** or **pg_num** attributes. This minimimzes data rebalancing.

To set the **target_size_ratio** attribute per pool, use a configuration file entry similar to the following example:

```
parameter_defaults:
  CephPools:
    - name: volumes
      target_size_ratio: 0.4
      application: rbd
    - name: images
      target_size_ratio: 0.1
      application: rbd
    - name: vms
      target_size_ratio: 0.3
      application: rbd
```

In this example, the percentage of data used per service will be:

- Cinder volumes - 40%
- Glance images - 10%
- Nova vms - 30%
- Free space for other pools - 20%

Set these values based on your expected usage . If you do not override the **CephPools** parameter, each pool uses the default placement group number. Though the autoscaler will adjust this number automatically over time based on usage, the data will be moved within the Ceph cluster. This uses computational resources.

If you prefer to set a placement group number instead of a target size ratio, replace **target_size_ratio** in the example with **pg_num**. Use a different integer per pool based on your expected usage.

See the Red Hat Ceph Storage Hardware Guide for Red Hat Ceph Storage processor, network interface card, and power management interface recommendations.

5.3. ENABLING CEPH METADATA SERVER

The Ceph Metadata Server (MDS) runs the **ceph-mds** daemon. This daemon manages metadata related to files stored on CephFS. CephFS can be consumed natively or through the NFS protocol.

NOTE

Red Hat supports deploying Ceph MDS with the native CephFS and CephFS NFS back ends for the Shared File Systems service (manila).

Procedure

- To enable Ceph MDS, use the following environment file when you deploy the overcloud:

```
/usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml
```

NOTE

By default, Ceph MDS is deployed on the Controller node. You can deploy Ceph MDS on its own dedicated node.

Additional resources

- Red Hat Ceph Storage *File System Guide*

5.4. CEPH OBJECT GATEWAY OBJECT STORAGE

The Ceph Object Gateway (RGW) provides an interface to access object storage capabilities within a Red Hat Ceph Storage cluster.

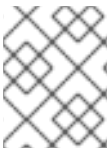
When you use director to deploy Ceph, director automatically enables RGW. This is a direct replacement for the Object Storage service (swift). Services that normally use the Object Storage service can use RGW instead without additional configuration. The Object Storage service remains available as an object storage option for upgraded Ceph clusters.

There is no requirement for a separate RGW environment file to enable it. For more information about environment files for other object storage options, see Section 5.5, "Deployment options for Red Hat OpenStack Platform object storage".

By default, Ceph Storage allows 250 placement groups per Object Storage Daemon (OSD). When you enable RGW, Ceph Storage creates the following six additional pools required by RGW:

- **.rgw.root**
- **<zone_name>.rgw.control**
- **<zone_name>.rgw.meta**

- **<zone_name>.rgw.log**
- **<zone_name>.rgw.buckets.index**
- **<zone_name>.rgw.buckets.data**



NOTE

In your deployment, **<zone_name>** is replaced with the name of the zone to which the pools belong.

Additional resources

- For more information about RGW, see the Red Hat Ceph Storage Object Gateway Guide .
- For more information about using RGW instead of Swift, see the Block Storage Backup Guide .

5.5. DEPLOYMENT OPTIONS FOR RED HAT OPENSTACK PLATFORM OBJECT STORAGE

There are three options for deploying overcloud object storage:

- Ceph Object Gateway (RGW)
  To deploy RGW as described in Section 5.4, "Ceph Object Gateway object storage" , include the following environment file during overcloud deployment:

  ```
  -e  environments/cephadm/cephadm.yaml
  ```

  This environment file configures both Ceph block storage (RBD) and RGW.

- Object Storage service (swift)
  To deploy the Object Storage service (swift) instead of RGW, include the following environment file during overcloud deployment:

  ```
  -e  environments/cephadm/cephadm-rbd-only.yaml
  ```

  The **cephadm-rbd-only.yaml** file configures Ceph RBD but not RGW.

  NOTE

  If you used the Object Storage service (swift) before upgrading your Red Hat Ceph Storage cluster, you can continue to use the Object Storage service (swift) instead of RGW by replacing the **environments/ceph-ansible/ceph-ansible.yaml** file with the **environments/cephadm/cephadm-rbd-only.yaml** during the upgrade. For more information, see Keeping Red Hat OpenStack Platform Updated.

  Red Hat OpenStack Platform does not support migration from the Object Storage service (swift) to Ceph Object Gateway (RGW).

- No object storage
  To deploy Ceph with RBD but not with RGW or the Object Storage service (swift), include the following environment files during overcloud deployment:

```
-e  environments/cephadm/cephadm-rbd-only.yaml
-e  environments/disable-swift.yaml
```

The **cephadm-rbd-only.yaml** file configures RBD but not RGW. The **disable-swift.yaml** file ensures that the Object Storage service (swift) does not deploy.

5.6. CONFIGURING THE BLOCK STORAGE BACKUP SERVICE TO USE CEPH

The Block Storage Backup service (cinder-backup) is disabled by default. It must be enabled to use it with Ceph.

Procedure

To enable the Block Storage Backup service (cinder-backup), use the following environment file when you deploy the overcloud:

```
`/usr/share/openstack-tripleo-heat-templates/environments/cinder-backup.yaml`.
```

5.7. CONFIGURING MULTIPLE BONDED INTERFACES FOR CEPH NODES

Use a bonded interface to combine multiple NICs and add redundancy to a network connection. If you have enough NICs on your Ceph nodes, you can create multiple bonded interfaces on each node to expand redundancy capability.

Use a bonded interface for each network connection the node requires. This provides both redundancy and a dedicated connection for each network.

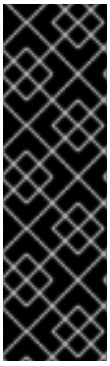
See Provisioning the overcloud networks in the *Director Installation and Usage* guide for information and procedures.

CHAPTER 6. DEPLOYING THE SHARED FILE SYSTEMS SERVICE WITH NATIVE CEPHFS

CephFS is the highly scalable, open-source, distributed file system component of Red Hat Ceph Storage, a unified distributed storage platform. Ceph Storage implements object, block, and file storage using Reliable Autonomic Distributed Object Store (RADOS). CephFS, which is POSIX compatible, provides file access to a Ceph Storage cluster.

The Shared File Systems service (manila) enables users to create shares in CephFS and access them using the native Ceph FS protocol. The Shared File Systems service manages the life cycle of these shares from within OpenStack.

With this release, director can deploy the Shared File Systems with a native CephFS back end on the overcloud.



IMPORTANT

This chapter pertains to the deployment and use of native CephFS to provide a self-service Shared File Systems service in your Red Hat OpenStack Platform(RHOSP) cloud through the native CephFS NAS protocol. This type of deployment requires guest VM access to Ceph public network and infrastructure. Deploy native CephFS with trusted OpenStack Platform tenants only, because it requires a permissive trust model that is not suitable for general purpose OpenStack Platform deployments. For general purpose OpenStack Platform deployments that use a conventional tenant trust model, you can deploy CephFS through the NFS protocol.

6.1. CEPHFS WITH NATIVE DRIVER

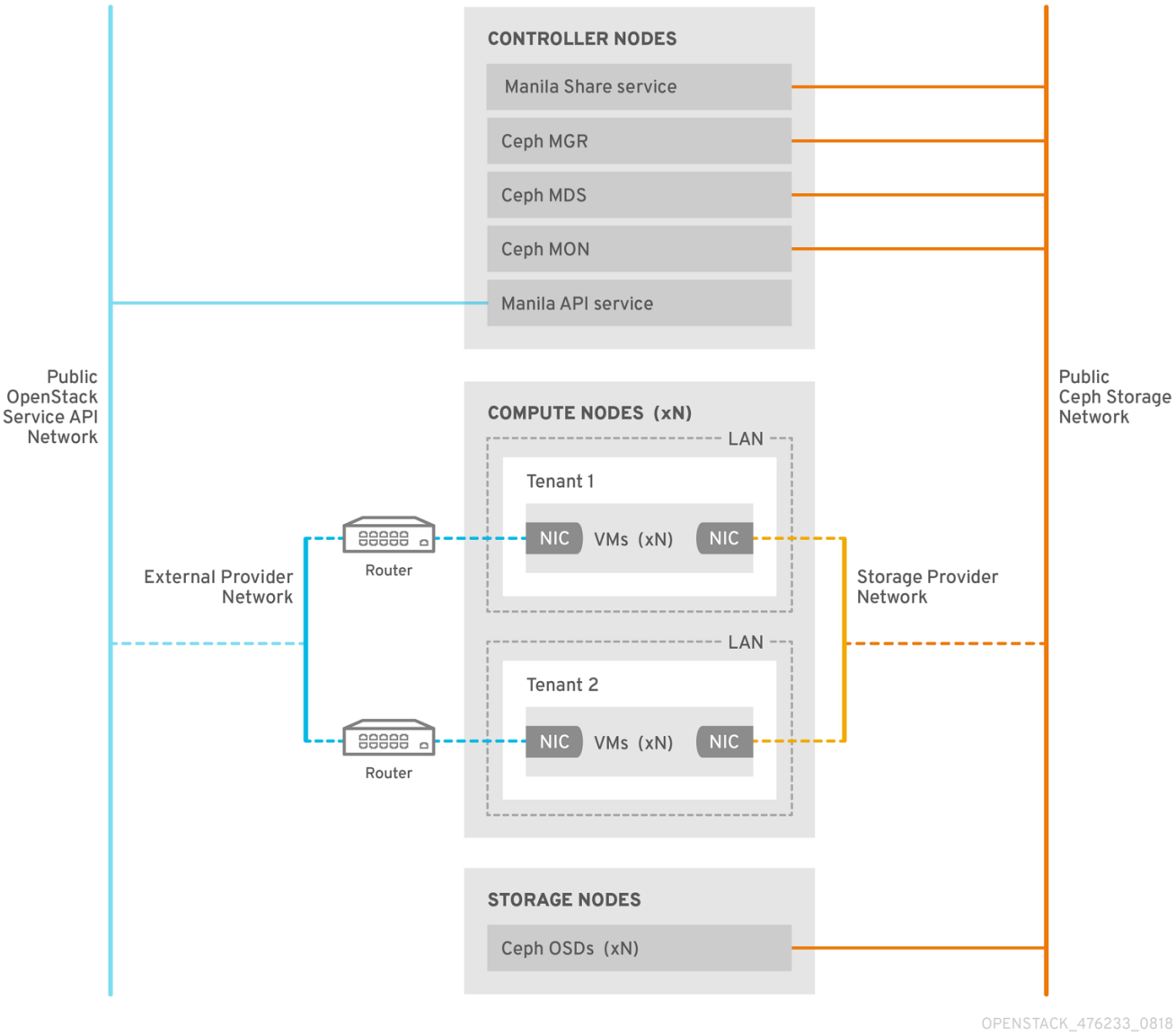
The CephFS native driver combines the OpenStack Shared File Systems service (manila) and Red Hat Ceph Storage. When you use Red Hat OpenStack (RHOSP) director, the Controller nodes host the Ceph daemons, such as the manager, metadata servers (MDS), and monitors (MON) and the Shared File Systems services.

Compute nodes can host one or more projects. Projects, which were formerly referred to as tenants, are represented in the following graphic by the white boxes. Projects contain user-managed VMs, which are represented by gray boxes with two NICs. To access the ceph and manila daemons projects, connect to the daemons over the public Ceph storage network.

On this network, you can access data on the storage nodes provided by the Ceph Object Storage Daemons (OSDs). Instances, or virtual machines (VMs), that are hosted on the project boot with two NICs: one dedicated to the storage provider network and the second to project-owned routers to the external provider network.

The storage provider network connects the VMs that run on the projects to the public Ceph storage network. The Ceph public network provides back-end access to the Ceph object storage nodes, metadata servers (MDS), and Controller nodes.

Using the native driver, CephFS relies on cooperation with the clients and servers to enforce quotas, guarantee project isolation, and for security. CephFS with the native driver works well in an environment with trusted end users on a private cloud. This configuration requires software that is running under user control to cooperate and work correctly.



6.2. NATIVE CEPHFS BACK-END SECURITY

The native CephFS back end requires a permissive trust model for Red Hat OpenStack Platform (RHOSP) tenants. This trust model is not appropriate for general purpose OpenStack Platform clouds that deliberately block users from directly accessing the infrastructure behind the services that the OpenStack Platform provides.

With native CephFS, user Compute instances connect directly to the Ceph public network where the Ceph service daemons are exposed. CephFS clients that run on user VMs interact cooperatively with the Ceph service daemons, and they interact directly with RADOS to read and write file data blocks.

CephFS quotas, which enforce Shared File Systems (manila) share sizes, are enforced on the client side, such as on VMs that are owned by (RHOSP) users. The client side software on user VMs might not be current, which can leave critical cloud infrastructure vulnerable to malicious or inadvertently harmful software that targets the Ceph service ports.

Deploy native CephFS as a back end only in environments in which trusted users keep client-side software up to date. Ensure that no software that can impact the Red Hat Ceph Storage infrastructure runs on your VMs.

For a general purpose RHOSP deployment that serves many untrusted users, deploy CephFS through NFS. For more information about using CephFS through NFS, see *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*.

Users might not keep client-side software current, and they might fail to exclude harmful software from their VMs, but using CephFS through NFS, they only have access to the public side of an NFS server, not to the Ceph infrastructure itself. NFS does not require the same kind of cooperative client and, in the worst case, an attack from a user VM can damage the NFS gateway without damaging the Ceph Storage infrastructure behind it.

You can expose the native CephFS back end to all trusted users, but you must enact the following security measures:

- Configure the storage network as a provider network.
- Impose role-based access control (RBAC) policies to secure the Storage provider network.
- Create a private share type.

6.3. NATIVE CEPHFS DEPLOYMENT

A typical native Ceph file system (CephFS) installation in a Red Hat OpenStack Platform (RHOSP) environment includes the following components:

- RHOSP Controller nodes that run containerized Ceph metadata server (MDS), Ceph monitor (MON) and Shared File Systems (manila) services. Some of these services can coexist on the same node or they can have one or more dedicated nodes.
- Ceph Storage cluster with containerized object storage daemons (OSDs) that run on Ceph Storage nodes.
- An isolated storage network that serves as the Ceph public network on which the clients can communicate with Ceph service daemons. To facilitate this, the storage network is made available as a provider network for users to connect their VMs and mount CephFS shares.

IMPORTANT

You cannot use the Shared File Systems service (manila) with the CephFS native driver to serve shares to OpenShift Container Platform through Manila CSI, because Red Hat does not support this type of deployment. For more information, contact Red Hat Support.

The Shared File Systems (manila) service provides APIs that allow the tenants to request file system shares, which are fulfilled by driver modules. The driver for Red Hat CephFS, **manila.share.drivers.cephfs.driver.CephFSDriver**, allows the Shared File Systems service to use native CephFS as a back end. You can install native CephFS in an integrated deployment managed by director.

When director deploys the Shared File Systems service with a CephFS back end on the overcloud, it automatically creates the required data center storage network. However, you must create the corresponding storage provider network on the overcloud.

For more information about network planning, see Overcloud networks in *Director Installation and Usage*.

Although you can manually configure the Shared File Systems service by editing the **/var/lib/config-data/puppet-generated/manila/etc/manila/manila.conf** file for the node, any settings can be

overwritten by the Red Hat OpenStack Platform director in future overcloud updates. Red Hat only supports deployments of the Shared File Systems service that are managed by director.

6.4. REQUIREMENTS

You can deploy a native CephFS back end with new or existing Red Hat OpenStack Platform (RHOSP) environments if you meet the following requirements:

- Use Red Hat OpenStack Platform version 17.0 or later.
- Configure a new Red Hat Ceph Storage cluster at the same time as the native CephFS back end. For information about how to deploy Ceph Storage, see *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*.

IMPORTANT

The RHOSP Shared File Systems service (manila) with the native CephFS back end is supported for use with Red Hat Ceph Storage version 5.2 or later. For more information about how to determine the version of Ceph Storage installed on your system, see Red Hat Ceph Storage releases and corresponding Ceph package versions.

- Install the Shared File Systems service on a Controller node. This is the default behavior.
- Use only a single instance of a CephFS back end for the Shared File Systems service.

6.5. FILE SHARES

File shares are handled differently between the Shared File Systems service (manila), Ceph File System (CephFS), and CephFS through NFS.

The Shared File Systems service provides shares, where a share is an individual file system namespace and a unit of storage with a defined size. Shared file system storage inherently allows multiple clients to connect, read, and write data to any given share, but you must give each client access to the share through the Shared File Systems service access control APIs before they can connect.

With CephFS, a share is considered a directory with a defined quota and a layout that points to a particular storage pool or namespace. CephFS quotas limit the size of a directory to the size share that the Shared File Systems service creates. Access to Ceph shares is determined by MDS authentication capabilities.

With native CephFS, file shares are provisioned and accessed through the CephFS protocol. Access control is performed with a CephX authentication scheme that uses CephFS usernames.

6.6. NATIVE CEPHFS ISOLATED NETWORK

Native CephFS deployments use the isolated storage network deployed by director as the Ceph public network. Clients use this network to communicate with various Ceph infrastructure service daemons. For more information about isolating networks, see Network isolation in *Director Installation and Usage*.

6.7. DEPLOYING THE NATIVE CEPHFS ENVIRONMENT

When you are ready to deploy the environment, use the **openstack overcloud deploy** command with the custom environments and roles required to configure the native CephFS back end.

The **openstack overcloud deploy** command has the following options in addition to other required options.

Action	Option	Additional Information
Specify the network configuration with **network_data.yaml**	[filename] **-n /usr/share/openstack-tripleo-heat-templates/network_data.yaml**	You can use a custom environment file to override values for the default networks specified in this network data environment file. This is the default network data file that is available when you use isolated networks. You can omit this file from the **openstack overcloud deploy** command for brevity.
Deploy the Ceph daemons.	**-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml**	Initiating overcloud deployment in *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*
Deploy the Ceph metadata server with **ceph-mds.yaml**	**-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml**	Initiating overcloud deployment in *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*
Deploy the manila service with the native CephFS back end.	**-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsnative-config.yaml**	Environment file

The following example shows an **openstack overcloud deploy command** that includes options to deploy a Ceph cluster, Ceph MDS, the native CephFS back end, and the networks required for the Ceph cluster:

```
[stack@undercloud ~]$ openstack overcloud deploy \
...
-n /usr/share/openstack-tripleo-heat-templates/network_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml   \
-e /home/stack/network-environment.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsnative-config.yaml
```

For more information about the **openstack overcloud deploy** command, see Provisioning and deploying your overcloud in *Director Installation and Usage*.

6.8. NATIVE CEPHFS BACK-END ENVIRONMENT FILE

The environment file for defining a native CephFS back end, **manila-cephfsnative-config.yaml** is located in the following path of an undercloud node: **/usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsnative-config.yaml**.

The **manila-cephfsnative-config.yaml** environment file contains settings relevant to the deployment of the Shared File Systems service. The back end default settings should work for most environments.

The example shows the default values that director uses during deployment of the Shared File Systems service:

```
[stack@undercloud ~]$ cat /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsnative-config.yaml

# A Heat environment file which can be used to enable a
# a Manila CephFS Native driver backend.
resource_registry:
  OS::TripleO::Services::ManilaApi: ../deployment/manila/manila-api-container-puppet.yaml
  OS::TripleO::Services::ManilaScheduler: ../deployment/manila/manila-scheduler-container-puppet.yaml
  # Only manila-share is pacemaker managed:
  OS::TripleO::Services::ManilaShare: ../deployment/manila/manila-share-pacemaker-puppet.yaml
  OS::TripleO::Services::ManilaBackendCephFs: ../deployment/manila/manila-backend-cephfs.yaml

parameter_defaults:
  ManilaCephFSBackendName: cephfs (1)
  ManilaCephFSDriverHandlesShareServers: false (2)
  ManilaCephFSCephFSAuthId: 'manila' (3)
  ManilaCephFSCephFSEnableSnapshots: true (4)
  ManilaCephFSCephVolumeMode: '0755' (5)
  # manila cephfs driver supports either native cephfs backend - 'CEPHFS'
  # (users mount shares directly from ceph cluster), or nfs-ganesha backend -
  # 'NFS' (users mount shares through nfs-ganesha server)
  ManilaCephFSCephFSProtocolHelperType: 'CEPHFS' (6)
```

The **parameter_defaults** header signifies the start of the configuration. Specifically, settings under this header let you override default values set in **resource_registry**. This includes values set by **OS::Tripleo::Services::ManilaBackendCephFs**, which sets defaults for a CephFS back end.

1. **ManilaCephFSBackendName** sets the name of the manila configuration of your CephFS backend. In this case, the default back end name is **cephfs**.

2. **ManilaCephFSDriverHandlesShareServers** controls the lifecycle of the share server. When set to **false**, the driver does not handle the lifecycle. This is the only supported option for CephFS back ends.

3. **ManilaCephFSCephFSAuthId** defines the Ceph auth ID that the director creates for the manila service to access the Ceph cluster.

4. **ManilaCephFSCephFSEnableSnapshots** controls snapshot activation. Snapshots are supported With Ceph Storage 4.1 and later, but the value of this parameter defaults to **false**. You can set the value to **true** to ensure that the driver reports the **snapshot_support** capability to the manila scheduler.

5. **ManilaCephFSCephVolumeMode** controls the UNIX permissions to set against the manila share created on the native CephFS back end. The value defaults to **755**.

6. **ManilaCephFSCephFSProtocolHelperType** must be set to **CEPHFS** to use the native CephFS driver.

For more information about environment files, see Environment Files in the *Director Installation and Usage* guide.

CHAPTER 7. DEPLOYING THE SHARED FILE SYSTEMS SERVICE WITH CEPHFS-NFS

With the Shared File Systems service (manila) with Ceph File System (CephFS) through NFS, you can use the same Red Hat Ceph Storage cluster that you use for block and object storage to provide file shares through the NFS protocol. For more information, see Configuring the Shared File Systems service (manila) in the *Storage Guide*.

IMPORTANT

The RHOSP Shared File Systems service with CephFS through NFS for RHOSP 17.0 and later is supported for use with Red Hat Ceph Storage version 5.x or later. For more information about how to determine the version of Ceph Storage installed on your system, see Red Hat Ceph Storage releases and corresponding Ceph package versions .

CephFS is the highly scalable, open-source distributed file system component of Red Hat Ceph Storage, a unified distributed storage platform. Ceph Storage implements object, block, and file storage using Reliable Autonomic Distributed Object Store (RADOS). CephFS, which is POSIX compatible, provides file access to a Ceph storage cluster.

The Shared File Systems service enables users to create shares in CephFS and access them with NFS 4.1 through NFS-Ganesha. NFS-Ganesha controls access to the shares and exports them to clients through the NFS 4.1 protocol.

The Shared File Systems service manages the life cycle of these shares from within RHOSP. When cloud administrators configure the service to use CephFS through NFS, these file shares come from the CephFS cluster, but are created and accessed as familiar NFS shares.

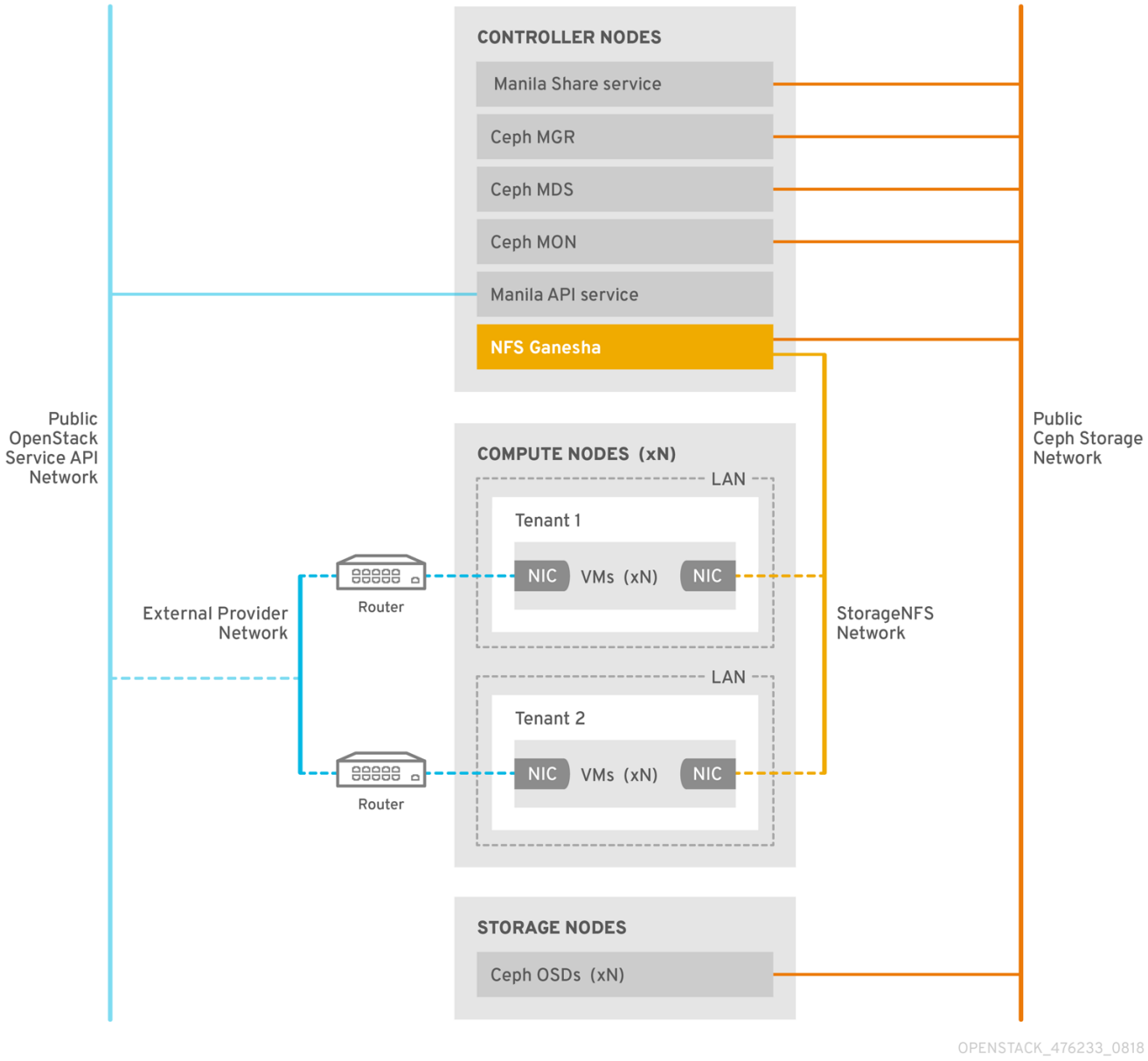
For more information about the Shared File Systems service, see Configuring the Shared File Systems service (manila) in the *Storage Guide*.

7.1. CEPHFS WITH NFS DRIVER

The CephFS through NFS back end in the Shared File Systems service (manila) is composed of Ceph metadata servers (MDS), the CephFS through NFS gateway (NFS-Ganesha), and the Ceph cluster service components. The Shared File Systems service CephFS NFS driver uses NFS-Ganesha gateway to provide NFSv4 protocol access to CephFS shares. The Ceph MDS service maps the directories and file names of the file system to objects that are stored in RADOS clusters. NFS gateways can serve NFS file shares with different storage back ends, such as Ceph. The NFS-Ganesha service runs on the Controller nodes with the Ceph services.

Instances are booted with at least two NICs: one NIC connects to the project router and the second NIC connects to the StorageNFS network, which connects directly to the NFS-Ganesha gateway. The instance mounts shares by using the NFS protocol. CephFS shares that are hosted on Ceph OSD nodes are provided through the NFS gateway.

NFS-Ganesha improves security by preventing user instances from directly accessing the MDS and other Ceph services. Instances do not have direct access to the Ceph daemons.



7.2. CEPH SERVICES AND CLIENT ACCESS

In addition to the monitor, OSD, Rados Gateway (RGW), and manager services deployed when Ceph provides object and block storage, a Ceph metadata service (MDS) is required for CephFS and an NFS-Ganesha service is required as a gateway to native CephFS using the NFS protocol. For user-facing object storage, an RGW service is also deployed. The gateway runs the CephFS client to access the Ceph public network and is under administrative rather than end-user control.

NFS-Ganesha runs in its own container that interfaces both to the Ceph public network and to a new isolated network, StorageNFS. The composable network feature of Red Hat OpenStack Platform (RHOSP) director deploys this network and connects it to the Controller nodes. As the cloud administrator, you can configure the network as a Networking (neutron) provider network.

NFS-Ganesha accesses CephFS over the Ceph public network and binds its NFS service using an address on the StorageNFS network.

To access NFS shares, provision user VMs, Compute (nova) instances, with an additional NIC that connects to the Storage NFS network. Export locations for CephFS shares appear as standard NFS **IP:<path>** tuples that use the NFS-Ganesha server VIP on the StorageNFS network. The network uses the

IP address of the user VM to perform access control on the NFS shares.

Networking (neutron) security groups prevent the user VM that belongs to project 1 from accessing a user VM that belongs to project 2 over the StorageNFS network. Projects share the same CephFS file system but project data path separation is enforced because user VMs can access files only under export trees: **/path/to/share1/...**, **/path/to/share2/...**.

7.3. SHARED FILE SYSTEMS SERVICE WITH CEPHFS THROUGH NFS FAULT TOLERANCE

When Red Hat OpenStack Platform (RHOSP) director starts the Ceph service daemons, they manage their own high availability (HA) state and, in general, there are multiple instances of these daemons running. By contrast, in this release, only one instance of NFS-Ganesha can serve file shares at a time.

To avoid a single point of failure in the data path for CephFS through NFS shares, NFS-Ganesha runs on a RHOSP Controller node in an active-passive configuration managed by a Pacemaker-Corosync cluster. NFS-Ganesha acts across the Controller nodes as a virtual service with a virtual service IP address.

If a Controller node fails or the service on a particular Controller node fails and cannot be recovered on that node, Pacemaker-Corosync starts a new NFS-Ganesha instance on a different Controller node using the same virtual IP address. Existing client mounts are preserved because they use the virtual IP address for the export location of shares.

Using default NFS mount-option settings and NFS 4.1 or later, after a failure, TCP connections are reset and clients reconnect. I/O operations temporarily stop responding during failover, but they do not fail. Application I/O also stops responding but resumes after failover completes.

New connections, new lock-state, and so on are refused until after a grace period of up to 90 seconds during which time the server waits for clients to reclaim their locks. NFS-Ganesha keeps a list of the clients and exits the grace period earlier if all clients reclaim their locks.

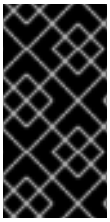
NOTE

The default value of the grace period is 90 seconds. To change this value, edit the NFSv4 **Grace_Period** configuration option.

7.4. CEPHFS THROUGH NFS-GANESHA INSTALLATION

A typical Ceph file system (CephFS) through NFS installation in a Red Hat OpenStack Platform (RHOSP) environment includes the following configurations:

- OpenStack Controller nodes running containerized Ceph metadata server (MDS), Ceph monitor (MON), manila, and NFS-Ganesha services. Some of these services can coexist on the same node or can have one or more dedicated nodes.
- Ceph storage cluster with containerized object storage daemons (OSDs) running on Ceph storage nodes.
- An isolated StorageNFS network that provides access from projects to the NFS-Ganesha services for NFS share provisioning.



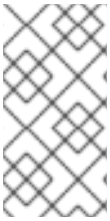
IMPORTANT

The Shared File Systems service (manila) with CephFS through NFS fully supports serving shares to Red Hat OpenShift Container Platform through Manila CSI. This solution is not intended for large scale deployments. For important recommendations, see https://access.redhat.com/articles/6667651.

The Shared File Systems service (manila) provides APIs that allow the projects to request file system shares, which are fulfilled by driver modules. The driver for Red Hat CephFS, **manila.share.drivers.cephfs.driver.CephFSDriver**, means that you can use the Shared File Systems service as a CephFS back end. RHOSP director configures the driver to deploy the NFS-Ganesha gateway so that the CephFS shares are presented through the NFS 4.1 protocol.

Using RHOSP director to deploy the Shared File Systems service with a CephFS back end on the overcloud automatically creates the required storage network defined in the heat template. For more information about network planning, see Overcloud networks in the *Director Installation and Usage* guide.

Although you can manually configure the Shared File Systems service by editing its node **/etc/manila/manila.conf** file, RHOSP director can override any settings in future overcloud updates. The recommended method for configuring a Shared File Systems back end is through director. Use RHOSP director to create an extra StorageNFS network for storage traffic.



NOTE

Adding CephFS through NFS to an externally deployed Ceph cluster, which was not configured by director, is supported. Currently, you can only define one CephFS back end in director. For more information, see Integrating an overcloud with Ceph Storage in *Integrating an Overcloud with an Existing Red Hat Ceph Storage Cluster* .

7.5. CEPHFS THROUGH NFS-GANESHA INSTALLATION REQUIREMENTS

CephFS through NFS has been fully supported since Red Hat OpenStack Platform version (RHOSP) 13. The RHOSP Shared File Systems service with CephFS through NFS for RHOSP 17.0 and later is supported for use with Red Hat Ceph Storage version 5.2 or later. For more information about how to determine the version of Ceph Storage installed on your system, see Red Hat Ceph Storage releases and corresponding Ceph package versions.

Prerequisites

- You install the Shared File Systems service on Controller nodes, as is the default behavior.
- You install the NFS-Ganesha gateway service on the Pacemaker cluster of the Controller node.
- You configure only a single instance of a CephFS back end to use the Shared File Systems service. You can use other non-CephFS back ends with the single CephFS back end.

7.6. FILE SHARES

File shares are handled differently between the OpenStack Shared File Systems service (manila), Ceph File System (CephFS), and Ceph through NFS.

The Shared File Systems service provides shares, where a share is an individual file system namespace

and a unit of storage with a defined size. Shared file system storage inherently allows multiple clients to connect, read, and write data to any given share, but you must give each client access to the share through the Shared File Systems service access control APIs before they can connect.

With CephFS, a share is considered a directory with a defined quota and a layout that points to a particular storage pool or namespace. CephFS quotas limit the size of a directory to the size share that the Shared File Systems service creates. Access to CephFS through NFS shares is provided by specifying the IP address of the client.

With CephFS through NFS, file shares are provisioned and accessed through the NFS protocol. The NFS protocol also handles security.

7.7. GENERATING THE CUSTOM ROLES FILE

For security, isolate NFS traffic to a separate network when using CephFS through NFS so that the Ceph NFS server is accessible only through the isolated network. Deployers can restrict the isolated network to a select group of projects in the cloud. Red Hat OpenStack (RHOSP) director ships with support to deploy a dedicated StorageNFS network. To configure and use the StorageNFS network, you require a custom Controller role.

IMPORTANT

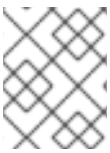
It is possible to omit the creation of an isolated network for NFS traffic. However, if you omit the StorageNFS network in a production deployment that has untrusted clients, director can connect the Ceph NFS server on any shared non-isolated network, such as an external network. Shared non-isolated networks are usually routable to all user private networks in the cloud. When the NFS server is on a non-isolated network, you cannot control access to Shared File Systems service (manila) shares by applying client IP access rules. Users must allow access to their shares by using the generic **0.0.0.0/0** IP. Because of the generic IP, anyone who discovers the export path can mount the shares.

You configure the isolated StorageNFS network by using the **ControllerStorageNFS** custom role file. This role file is similar to the default **Controller.yaml** role file, with the addition of the **StorageNFS** network.

```
[stack@undercloud ~]$ cd /usr/share/openstack-tripleo-heat-templates/roles
[stack@undercloud roles]$ diff Controller.yaml ControllerStorageNfs.yaml
16a17
>   - StorageNFS
50a45
```

The **openstack overcloud roles generate** command creates a custom **roles_data.yaml** file, including the services specified after **-o**. In the following example, the **roles_data.yaml** file created has the services for **ControllerStorageNfs**, **Compute**, and **CephStorage**.

For more information about the **openstack overcloud roles generate** command, see Composable services and custom roles in *Director Installation and Usage*.



NOTE

If you have an existing **roles_data.yaml** file, modify it to add **ControllerStorageNfs**, **Compute**, and **CephStorage** services to the configuration file.

Procedure

1. Log in to the undercloud host as the **stack** user.
2. Source the **stackrc** undercloud credentials file:

```
$ source ~/stackrc
```

3. Create the **roles_data.yaml** file:

```
[stack@undercloud ~]$ openstack overcloud roles generate \
--roles-path /usr/share/openstack-tripleo-heat-templates/roles \
-o /home/stack/roles_data.yaml ControllerStorageNfs Compute CephStorage
```

7.8. DEPLOYING THE UPDATED ENVIRONMENT

When you are ready to deploy your environment, use the **openstack overcloud deploy** command with the custom environments and roles required to run CephFS with NFS-Ganesha.

The overcloud deploy command has the following options in addition to other required options.

Action	Option	Additional information
Add the extra StorageNFS network with **network_data_ganesha.yaml**	**-n /usr/share/openstack-tripleo-heat-templates/network_data_ganesha.yaml**	You can omit this option if you do not want to isolate NFS traffic to a separate network.
Add the custom roles defined in the **roles_data.yaml** file from the previous section	**-r /home/stack/roles_data.yaml**	You can omit this option if you do not want to isolate NFS traffic to a separate network.
Deploy the Ceph daemons.	**-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml**	Initiating overcloud deployment in *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*
Deploy the Ceph metadata server with **ceph-mds.yaml**	**-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml**	Initiating overcloud deployment in *Deploying Red Hat Ceph Storage and Red Hat OpenStack Platform together with director*
Deploy the Shared File Systems service (manila) with the CephFS-NFS back end. Configure NFS-Ganesha with director.	**-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml**	The manila-cephfsganesha-config.yaml environment file

The following example shows an **openstack overcloud deploy command** with options to deploy CephFS through NFS-Ganesha, Ceph Storage cluster, Ceph MDS, and the isolated StorageNFS network:

```
[stack@undercloud ~]$ openstack overcloud deploy \
```

```
--templates /usr/share/openstack-tripleo-heat-templates  \
-n /usr/share/openstack-tripleo-heat-templates/network_data_ganesha.yaml \
-r /home/stack/roles_data.yaml \
-e /home/stack/containers-default-parameters.yaml   \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml   \
-e /home/stack/network-environment.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml
```

For more information about the **openstack overcloud deploy** command, see Provisioning and deploying your overcloud in *Director Installation and Usage*.

7.8.1. The StorageNFS and network_data_ganesha.yaml file

Use composable networks to define custom networks and assign them to any role. Instead of using the standard **network_data.yaml** file, you can configure the StorageNFS composable network with the **network_data_ganesha.yaml** file. Both of these roles are available in the **/usr/share/openstack-tripleo-heat-templates** directory.

IMPORTANT

If you do not define the Storage NFS network, director defaults to the external network. Although the external network can be useful in test and prototype environments, security on the external network is not sufficient for production environments. For example, if you expose the NFS service on the external network, a denial of service (DoS) attack can disrupt controller API access to all cloud users, not only consumers of NFS shares. By contrast, when you deploy the NFS service on a dedicated Storage NFS network, potential DoS attacks can target only NFS shares in the cloud. In addition to potential security risks, when you deploy the NFS service on an external network, additional routing configurations are required for precise access control to shares. On the Storage NFS network, however, you can use the client IP address on the network to achieve precise access control.

The **network_data_ganesha.yaml** file contains an additional section that defines the isolated StorageNFS network. Although the default settings work for most installations, you must edit the YAML file to add your network settings, including the VLAN ID, subnet, and other settings.

```
name: StorageNFS
enabled: true
vip: true
name_lower: storage_nfs
vlan: 70
ip_subnet: '172.17.0.0/20'
allocation_pools: [{'start': '172.17.0.4', 'end': '172.17.0.250'}]
ipv6_subnet: 'fd00:fd00:fd00:7000::/64'
ipv6_allocation_pools: [{'start': 'fd00:fd00:fd00:7000::4', 'end': 'fd00:fd00:fd00:7000::fffe'}]
```

For more information about composable networks, see Composable networks in *Director Installation and Usage*.

7.8.2. CephFS-NFS back-end environment file

The environment file for defining a CephFS-NFS back end, **manila-cephfsganesha-config.yaml**, is located in the following path of an undercloud node: **/usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml**.

The **manila-cephfsganesha-config.yaml** environment file contains settings relevant to the deployment of the Shared File Systems service (manila). The back-end default settings work for most environments. The following example shows the default values that director uses during deployment of the Shared File Systems service:

```
[stack@undercloud ~]$ cat /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml
# A Heat environment file which can be used to enable a
# a Manila CephFS-NFS driver backend.
resource_registry:
  OS::TripleO::Services::ManilaApi: ../deployment/manila/manila-api-container-puppet.yaml
  OS::TripleO::Services::ManilaScheduler: ../deployment/manila/manila-scheduler-container-puppet.yaml
  # Only manila-share is pacemaker managed:
  OS::TripleO::Services::ManilaShare: ../deployment/manila/manila-share-pacemaker-puppet.yaml
  OS::TripleO::Services::ManilaBackendCephFs: ../deployment/manila/manila-backend-cephfs.yaml
  # ceph-nfs (ganesha) service is installed and configured by Director
  # but it's still managed by pacemaker
  OS::TripleO::Services::CephNfs: ../deployment/cephadm/ceph-nfs.yaml

parameter_defaults:
  ManilaCephFSBackendName: cephfs (1)
  ManilaCephFSDriverHandlesShareServers: false (2)
  ManilaCephFSCephFSAuthId: 'manila' (3)
  # manila cephfs driver supports either native cephfs backend - 'CEPHFS'
  # (users mount shares directly from ceph cluster), or nfs-ganesha backend -
  # 'NFS' (users mount shares through nfs-ganesha server)
  ManilaCephFSCephFSProtocolHelperType: 'NFS'
```

The **parameter_defaults** header signifies the start of the configuration. To override default values set in **resource_registry**, copy this **manila-cephfsganesha-config.yaml** environment file to your local environment file directory, **/home/stack/templates/**, and edit the parameter settings as required by your environment. This includes values set by **OS::Tripleo::Services::ManilaBackendCephFs**, which sets defaults for a CephFS back end.

1. **ManilaCephFSBackendName** sets the name of the manila configuration of your CephFS back end. In this case, the default back-end name is **cephfs**.
2. **ManilaCephFSDriverHandlesShareServers** controls the lifecycle of the share server. When set to **false**, the driver does not handle the lifecycle. This is the only supported option.
3. **ManilaCephFSCephFSAuthId** defines the Ceph auth ID that director creates for the **manila** service to access the Ceph cluster.

For more information about environment files, see Environment files in *Director Installation and Usage*.

CHAPTER 8. INITIATING OVERCLOUD DEPLOYMENT

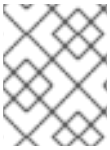
Deploy the overcloud after completing the initial configuration and customization of services.

8.1. INITIATING OVERCLOUD DEPLOYMENT

Deploy the overcloud to implement the configuration of the Red Hat OpenStack Platform (RHOSP) environment.

Prerequisites

- During undercloud installation, set **generate_service_certificate=false** in the **undercloud.conf** file. Otherwise, you must inject a trust anchor when you deploy the overcloud.



NOTE

If you want to add Ceph Dashboard during your overcloud deployment, see Chapter 10, *Adding the Red Hat Ceph Storage Dashboard to an overcloud deployment* .

Procedure

Deploy the overcloud using the **openstack overcloud deploy** command. For a complete list of all command arguments, see **openstack overcloud deploy** in the *Command Line Interface Reference* .

The following is an example usage of the command:

```
$ openstack overcloud deploy --templates -r /home/stack/templates/roles_data_custom.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cinder-backup.yaml \
  -e /home/stack/templates/storage-config.yaml \
  -e /home/stack/templates/deployed-ceph.yaml \
  -e /home/stack/templates/networks-deployed.yaml \
  -e /home/stack/templates/deployed-metal.yaml \
  -e /home/stack/templates/deployed-vips.yaml \
  --ntp-server pool.ntp.org
```

The example command uses the following options:

- **--templates**
  - Creates the overcloud from the default heat template collection, **/usr/share/openstack-tripleo-heat-templates/**.
- **-r /home/stack/templates/roles_data_custom.yaml**
  - Specifies a customized roles definition file.
- **-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml**
  - Sets the director to finalize the previously deployed Ceph Storage cluster. This environment file deploys RGW by default. It also creates pools, keys, and daemons. If you do not want to deploy RGW or object storage, see the options described in Section 5.5, "Deployment options for Red Hat OpenStack Platform object storage"

- **-e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-mds.yaml**
  - Enables the Ceph Metadata Server, as described in Section 5.3, "Enabling Ceph Metadata Server".
- **-e /usr/share/openstack-tripleo-heat-templates/environments/cinder-backup.yaml**
  - Enables the Block Storage Backup service (**cinder-backup**), as described in Section 5.6, "Configuring the Block Storage Backup Service to use Ceph".
- **-e /home/stack/templates/storage-config.yaml**
  - Adds the environment file that contains your custom Ceph Storage configuration as described in Section 5.1, "Configuring a custom environment file"
- **-e /home/stack/templates/deployed-ceph.yaml**
  - Adds the environment file that contains your Ceph cluster settings, as output by the **openstack overcloud ceph deploy** command run earlier.
- **-e /home/stack/templates/networks-deployed.yaml**
  - Adds the environment file that contains your Ceph cluster network settings, as output by **openstack overcloud network provision**.
- **-e /home/stack/templates/deployed-metal.yaml**
  - Adds the environment file that contains your Ceph cluster node settings, as output by **openstack overcloud node provision**.
- **-e /home/stack/templates/deployed-vips.yaml**
  - Adds the environment file that contains your Ceph cluster network VIP settings, as output by **openstack overcloud network vip provision**.
- **--ntp-server pool.ntp.org**
  - Sets the NTP server.

CHAPTER 9. USING DIRECTOR TO DEFINE PERFORMANCE TIERS FOR VARYING WORKLOADS

Red Hat OpenStack Platform (RHOSP) director deploys Red Hat Ceph Storage performance tiers. Ceph Storage CRUSH rules combine with the **CephPools** parameter to use the device classes features. This builds different tiers to accommodate workloads with different performance requirements.

For example, you can define a HDD class for normal workloads and an SSD class that distributes data only over SSDs for high performance loads. In this scenario, when you create a new Block Storage volume, you can choose the performance tier, either HDDs or SSDs.

For more information on CRUSH rule creation, see Configuring CRUSH hierarchies.

WARNING

Defining performance tiers in an existing environment can result in data movement in the Ceph Storage cluster. Director uses **cephadm** during the stack update. The **cephadm** application does not have the logic to verify if a pool exists and contains data. Changing the default CRUSH rule associated with a pool results in data movement. If the pool contains a large amount of data, that data will be moved.

If you require assistance or recommendations for adding or removing nodes, contact Red Hat support.

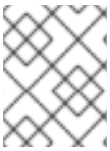
Ceph Storage automatically detects the disk type and assigns it to the corresponding device class; either HDD, SSD, or NVMe; based on the hardware properties exposed by the Linux kernel.

Prerequisites

- For new deployments, use Red Hat Ceph Storage (RHCS) version 5.2 or later.

9.1. CONFIGURING PERFORMANCE TIERS

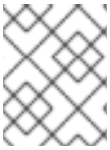
To deploy different Red Hat Ceph Storage performance tiers, create a new environment file that contains the CRUSH map details and include it in the deployment command. Director does not expose specific parameters for this feature, but you can generate the **tripleo-ansible** expected variables.



NOTE

Performance tier configuration can be combined with CRUSH hierarchies. See Configuring CRUSH hierarchies for information on CRUSH rule creation.

In the example procedure, each Ceph Storage node contains three OSDs: **sdb** and **sdc** are spinning disks and **sdc** is an SSD. Ceph automatically detects the correct disk type. You then configure two CRUSH rules, HDD and SSD, to map to the two respective device classes.



NOTE

The HDD rule is the default and applies to all pools unless you configure pools with a different rule.

Finally, you create an extra pool called **fastpool** and map it to the SSD rule. This pool is ultimately exposed through a Block Storage (cinder) back end. Any workload that consumes this Block Storage back end is backed by SSD for fast performances only. You can leverage this for either data or boot from volume.

WARNING

Defining performance tiers in an existing environment might result in massive data movement in the Ceph cluster. **cephadm**, which director triggers during the stack update, does not have logic to verify whether a pool is already defined in the Ceph cluster and if it contains data. This means that defining performance tiers in an existing environment can be dangerous because the change of the default CRUSH rule that is associated with a pool results in data movement. If you require assistance or recommendations for adding or removing nodes, contact Red Hat support.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Create an environment file, such as **/home/stack/templates/ceph-config.yaml**, to contain the Ceph config parameters and the device classes variables. Alternatively, you can add the following configurations to an existing environment file.
3. Add the **CephCrushRules** parameters. The **crush_rules** parameter must contain a rule for each class that you define or that Ceph detects automatically. When you create a new pool, if no rule is specified, the rule that you want Ceph to use as the default is selected.

```
CephCrushRules:
  crush_rules:
   - name: HDD
     root: default
     type: host
     class: hdd
     default: true
   - name: SSD
     root: default
     type: host
     class: ssd
     default: false
```

4. Add the **CephPools** parameter:
   - Use the **rule_name** parameter to specify the tier for each pool that does not use the default rule. In the following example, the **fastpool** pool uses the SSD device class that is configured as a fast tier, to manage Block Storage volumes.
   - Use the **CinderRbdExtraPools** parameter to configure **fastpool** as a Block Storage back end.

```
CephPools:
  - name: fastpool
    rule_name: SSD
```

```
    application: rbd
  CinderRbdExtraPools: fastpool
```

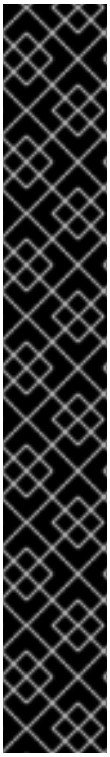
5. Use the following example to ensure that your environment file contains the correct values:

```
parameter_defaults:
      crush_rules:
        - name: HDD
          root: default
          type: host
          class: hdd
          default: true
        - name: SSD
          root: default
          type: host
          class: ssd
          default: false
  CinderRbdExtraPools: fastpool
  CephPools:
    - name: fastpool
      rule_name: SSD
      application: rbd
```

6. Include the new environment file in the **openstack overcloud deploy** command.

```
$ openstack overcloud deploy \
--templates \
…
-e <other_overcloud_environment_files> \
-e /home/stack/templates/ceph-config.yaml  \
…
```

Replace **<other_overcloud_environment_files>** with the list of other environment files that are part of your deployment.



IMPORTANT

If you apply the environment file to an existing Ceph cluster, the pre-existing Ceph pools are not updated with the new rules. For this reason, you must enter the following command after the deployment completes to set the rules to the specified pools.

```
$ ceph osd pool set <pool> crush_rule <rule>
```

- Replace <pool> with the name of the pool that you want to apply the new rule to.
- Replace <rule> with one of the rule names that you specified with the **crush_rules** parameter.

For every rule that you change with this command, update the existing entry or add a new entry in the **CephPools** parameter in your existing templates:

```
CephPools:
  - name: <pool>
    rule_name: <rule>
    application: rbd
```

9.2. VERIFYING CRUSH RULES AND POOLS

Verify your CRUSH rules and pools settings.

WARNING

Defining performance tiers in an existing environment might result in massive data movement in the Ceph cluster. **tripleo-ansible**, which director triggers during the stack update, does not have logic to check if a pool is already defined in the Ceph cluster and if it contains data. This means that defining performance tiers in an existing environment can be dangerous because the change of the default CRUSH rule that is associated with a pool results in data movement. If you require assistance or recommendations for adding or removing nodes, contact Red Hat support.

Procedure

1. Log in to the overcloud Controller node as the **tripleo-admin** user.
2. To verify that your OSD tiers are successfully set, enter the following command.

```
$ sudo cephadm shell ceph osd tree
```

3. In the resulting tree view, verify that the **CLASS** column displays the correct device class for each OSD that you set.
4. Also verify that the OSDs are correctly assigned to the device classes with the following command.

```
$ sudo cephadm shell ceph osd crush tree --show-shadow
```

5. Compare the resulting hierarchy with the results of the following command to ensure that the same values apply for each rule.

```
$ sudo cephadm shell ceph osd crush rule dump <rule_name>
```

- Replace <rule_name> with the name of the rule you want to check.

6. Verify that the rules name and ID that you created are correct according to the **crush_rules** parameter that you used during deployment.

```
$ sudo cephadm shell ceph osd crush rule dump | grep -E "rule_(id|name)"
```

7. Verify that the Ceph pools are tied to the correct CRUSH rule ID that you retrieved in Step 3.

```
$ sudo cephadm shell -- ceph osd dump | grep pool
```

8. For each pool, ensure that the rule ID matches the rule name that you expect.

CHAPTER 10. ADDING THE RED HAT CEPH STORAGE DASHBOARD TO AN OVERCLOUD DEPLOYMENT

Red Hat Ceph Storage Dashboard is disabled by default but you can enable it in your overcloud with the Red Hat OpenStack Platform (RHOSP) director. The Ceph Dashboard is a built-in, web-based Ceph management and monitoring application that administers various aspects and objects in your Ceph cluster. Red Hat Ceph Storage Dashboard comprises the following components:

- The Ceph Dashboard manager module provides the user interface and embeds the platform front end, Grafana.
- Prometheus, the monitoring plugin.
- Alertmanager sends alerts to the Dashboard.
- Node Exporters export Ceph cluster data to the Dashboard.

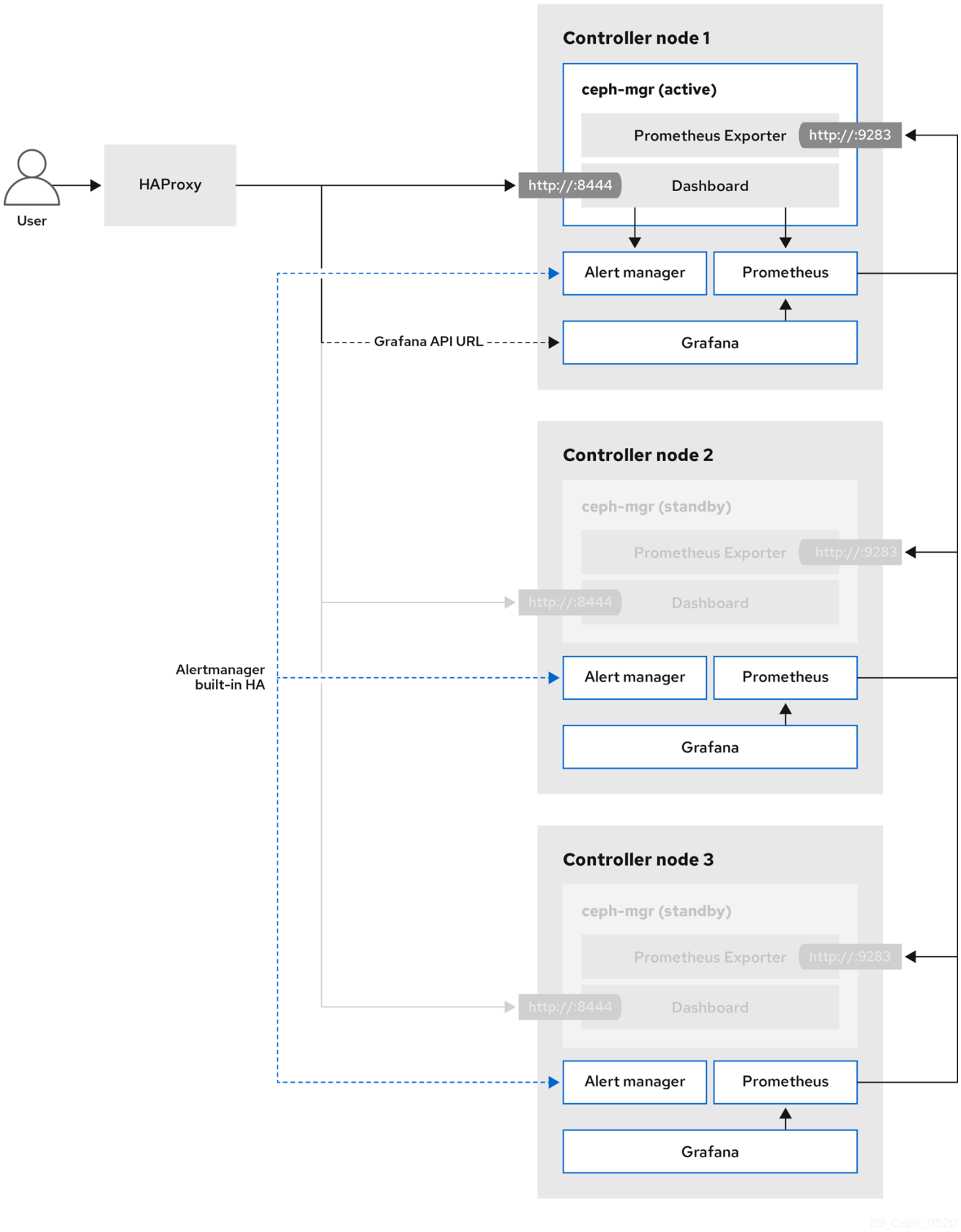
NOTE

This feature is supported with Ceph Storage 4.1 or later. For more information about how to determine the version of Ceph Storage installed on your system, see Red Hat Ceph Storage releases and corresponding Ceph package versions.

NOTE

The Red Hat Ceph Storage Dashboard is always colocated on the same nodes as the other Ceph manager components.

The following diagram shows the architecture of Ceph Dashboard on Red Hat OpenStack Platform:



For more information about the Dashboard and its features and limitations, see Dashboard features in the *Red Hat Ceph Storage Dashboard Guide* .

10.1. TLS EVERYWHERE WITH CEPH DASHBOARD

The Dashboard front end is fully integrated with the TLS everywhere framework. You can enable TLS

everywhere provided that you have the required environment files and they are included in the overcloud deploy command. This triggers the certificate request for both Grafana and the Ceph Dashboard and the generated certificate and key files are passed to **cephadm** during the overcloud deployment. For instructions and more information about how to enable TLS for the Dashboard as well as for other RHOSP services, see the following topics in the *Advanced Overcloud Customization* guide:

NOTE

The port to reach the Ceph Dashboard remains the same even in the TLS-everywhere context.

10.2. INCLUDING THE NECESSARY CONTAINERS FOR THE CEPH DASHBOARD

Before you can add the Ceph Dashboard templates to your overcloud, you must include the necessary containers by using the **containers-prepare-parameter.yaml** file. To generate the **containers-prepare-parameter.yaml** file to prepare your container images, complete the following steps:

Procedure

1. Log in to your undercloud host as the **stack** user.
2. Generate the default container image preparation file:

```
$ sudo openstack tripleo container image prepare default \
  --local-push-destination \
  --output-env-file containers-prepare-parameter.yaml
```

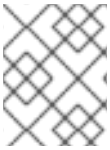
3. Edit the **containers-prepare-parameter.yaml** file and make the modifications to suit your requirements. The following example **containers-prepare-parameter.yaml** file contains the image locations and tags related to the Dashboard services including Grafana, Prometheus, Alertmanager, and Node Exporter. Edit the values depending on your specific scenario:

```
parameter_defaults:
    ContainerImagePrepare:
    - push_destination: true
        set:
            ceph_alertmanager_image: ose-prometheus-alertmanager
            ceph_alertmanager_namespace: registry.redhat.io/openshift4
            ceph_alertmanager_tag: v4.1
            ceph_grafana_image: rhceph-5-dashboard-rhel8
            ceph_grafana_namespace: registry.redhat.io/rhceph
            ceph_grafana_tag: 4
            ceph_image: rhceph-5-rhel8
            ceph_namespace: registry.redhat.io/rhceph
            ceph_node_exporter_image: ose-prometheus-node-exporter
            ceph_node_exporter_namespace: registry.redhat.io/openshift4
            ceph_node_exporter_tag: v4.1
            ceph_prometheus_image: ose-prometheus
            ceph_prometheus_namespace: registry.redhat.io/openshift4
            ceph_prometheus_tag: v4.1
            ceph_tag: latest
```

For more information about registry and image configuration with the **containers-prepare-parameter.yaml** file, see Container image preparation parameters in the *Transitioning to Containerized Services* guide.

10.3. DEPLOYING CEPH DASHBOARD

Include the ceph-dashboard environment file to deploy the Ceph Dashboard.



NOTE

If you want to deploy Ceph Dashboard with a composable network, see Section 10.4, "Deploying Ceph Dashboard with a composable network".

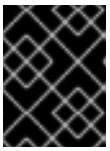
NOTE

The Ceph Dashboard admin user role is set to read-only mode by default. To change the Ceph Dashboard admin default mode, see Section 10.5, "Changing the default permissions".

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Optional: The Ceph Dashboard network is set by default to the provisioning network. If you want to deploy the Ceph Dashboard and access it through a different network, create an environment file, for example: **ceph_dashboard_network_override.yaml**. Set **CephDashboardNetwork** to one of the existing overcloud routed networks, for example **external**:

```
parameter_defaults:
  ServiceNetMap:
    CephDashboardNetwork: external
```



IMPORTANT

Changing the **CephDashboardNetwork** value to access the Ceph Dashboard from a different network is not supported after the initial deployment.

3. Include the following environment files in the **openstack overcloud deploy** command. Include all environment files that are part of your deployment, and the **ceph_dashboard_network_override.yaml** file if you chose to change the default network:

```
$ openstack overcloud deploy \
 --templates \
 -e <overcloud_environment_files> \
 -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml \
 -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-
dashboard.yaml \
 -e ceph_dashboard_network_override.yaml
```

Replace **<overcloud_environment_files>** with the list of environment files that are part of your deployment.

Result

The resulting deployment comprises an external stack with the grafana, prometheus, alertmanager, and node-exporter containers. The Ceph Dashboard manager module is the back end for this stack, and it embeds the grafana layouts to provide Ceph cluster specific metrics to the end users.

10.4. DEPLOYING CEPH DASHBOARD WITH A COMPOSABLE NETWORK

You can deploy the Ceph Dashboard on a composable network instead of on the default Provisioning network. This eliminates the need to expose the Ceph Dashboard service on the Provisioning network. When you deploy the Dashboard on a composable network, you can also implement separate authorization profiles.

You must choose which network to use before you deploy because you can apply the Dashboard to a new network only when you first deploy the overcloud. Use the following procedure to choose a composable network before you deploy.

Procedure

1. Log in to the undercloud as the stack user.
2. Generate the Controller specific role to include the Dashboard composable network:

```
$ openstack overcloud roles generate -o /home/stack/roles_data_dashboard.yaml
ControllerStorageDashboard Compute BlockStorage ObjectStorage CephStorage
```

Result

- A new **ControllerStorageDashboard** role is generated inside the **roles_data.yaml** defined as the output of the command. You must include this file in the template list when you use the overcloud deploy command.
  NOTE: The **ControllerStorageDashboard** role does not contain **CephNFS** nor **network_data_dashboard.yaml**.
- Director provides a network environment file where the composable network is defined. The default location of this file is **/usr/share/openstack-tripleo-heat-templates/network_data_dashboard.yaml**. You must include this file in the overcloud template list when you use the overcloud deploy command.

3. Include the following environment files, with all environment files that are part of your deployment, in the **openstack overcloud deploy** command:

```
$ openstack overcloud deploy \
  --templates \
  -r /home/stack/roles_data.yaml \
  -n /usr/share/openstack-tripleo-heat-templates/network_data_dashboard.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-environment.yaml \
  -e <overcloud_environment_files> \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/ceph-
dashboard.yaml
```

Replace **<overcloud_environment_files>** with the list of environment files that are part of your deployment.

Result

The resulting deployment comprises an external stack with the grafana, prometheus, alertmanager, and node-exporter containers. The Ceph Dashboard manager module is the back end for this stack, and it embeds the grafana layouts to provide Ceph cluster-specific metrics to the end users.

10.5. CHANGING THE DEFAULT PERMISSIONS

The Ceph Dashboard admin user role is set to read-only mode by default for safe monitoring of the Ceph cluster. To permit an admin user to have elevated privileges so that they can alter elements of the Ceph cluster with the Dashboard, you can use the **CephDashboardAdminRO** parameter to change the default admin permissions.

WARNING

A user with full permissions might alter elements of your Ceph cluster that director configures. This can cause a conflict with director-configured options when you run a stack update. To avoid this problem, do not alter director-configured options with Ceph Dashboard, for example, Ceph OSP pools attributes.

Procedure

1. Log in to the undercloud as the **stack** user.
2. Create the following **ceph_dashboard_admin.yaml** environment file:

```
parameter_defaults:
  CephDashboardAdminRO: false
```

3. Run the overcloud deploy command to update the existing stack and include the environment file you created with all other environment files that are part of your existing deployment:

```
$ openstack overcloud deploy \
--templates \
-e <existing_overcloud_environment_files> \
-e ceph_dashboard_admin.yml
```

Replace **<existing_overcloud_environment_files>** with the list of environment files that are part of your existing deployment.

10.6. ACCESSING CEPH DASHBOARD

To test that Ceph Dashboard is running correctly, complete the following verification steps to access it and check that the data it displays from the Ceph cluster is correct.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Retrieve the dashboard admin login credentials:

```
[stack@undercloud ~]$ grep tripleo_cephadm_dashboard_admin_password <config-download>/<stack>/cephadm/cephadm-extra-vars-heat.yml
```

3. Retrieve the VIP address to access the Ceph Dashboard:

```
[stack@undercloud-0 ~]$ grep tripleo_cephadm_dashboard_frontend_vip <config-download>/<stack>/cephadm/cephadm-extra-vars-ansible.yml
```

4. Use a web browser to point to the front end VIP and access the Dashboard. Director configures and exposes the Dashboard on the provisioning network, so you can use the VIP that you retrieved to access the Dashboard directly on TCP port 8444. Ensure that the following conditions are met:
   - The Web client host is layer 2 connected to the provisioning network.
   - The provisioning network is properly routed or proxied, and it can be reached from the web client host. If these conditions are not met, you can still open a SSH tunnel to reach the Dashboard VIP on the overcloud:

     ```
     client_host$ ssh -L 8444:<dashboard_vip>:8444 stack@<your undercloud>
     ```

     Replace <dashboard_vip> with the IP address of the control plane VIP that you retrieved.
5. To access the Dashboard, go to: http://localhost:8444 in a web browser and log in with the following details:
   - The default user that **cephadm** creates: **admin**.
   - The password in **<config-download>/<stack>/cephadm/cephadm-extra-vars-heat.yml**.

     Results

   - You can access the Ceph Dashboard.
   - The numbers and graphs that the Dashboard displays reflect the same Ceph cluster status that the CLI command, **ceph -s**, returns.

For more information about the Red Hat Ceph Storage Dashboard, see the *Red Hat Ceph Storage Administration Guide*

CHAPTER 11. POST-DEPLOYMENT OPERATIONS TO MANAGE THE RED HAT CEPH STORAGE CLUSTER

After you deploy your Red Hat OpenStack Platform (RHOSP) environment with containerized Red Hat Ceph Storage, there are some operations you can use to manage the Ceph Storage cluster.

11.1. DISABLING CONFIGURATION OVERRIDES

After the Ceph Storage cluster is initially deployed, the cluster is configured to allow the setup of services such as RGW during the overcloud deployment. Once overcloud deployment is complete, director should not be used to make changes to the cluster configuration unless you are scaling up the cluster. Cluster configuration changes should be performed using Ceph commands.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Open the file **deployed_ceph.yaml** or the file you use in your environment to define the Ceph Storage cluster configuration.
3. Locate the **ApplyCephConfigOverridesOnUpdate** parameter.
4. Change the **ApplyCephConfigOverridesOnUpdate** parameter value to **false**.
5. Save the file.

Additional resources

For more information on the **ApplyCephConfigOverridesOnUpdate** and **CephConfigOverrides** parameters, see Overcloud Parameters.

11.2. ACCESSING THE OVERCLOUD

Director generates a script to configure and help authenticate interactions with your overcloud from the undercloud. Director saves this file, **overcloudrc**, in the home directory of the **stack** user.

Procedure

1. Run the following command to source the file:

```
$ source ~/overcloudrc
```

   This loads the necessary environment variables to interact with your overcloud from the undercloud CLI.

2. To return to interacting with the undercloud, run the following command:

```
$ source ~/stackrc
```

11.3. MONITORING RED HAT CEPH STORAGE NODES

After you create the overcloud, check the status of the Ceph cluster to confirm that it works correctly.

Procedure

1. Log in to a Controller node as the **tripleo-admin** user:

```
$ nova list
$ ssh tripleo-admin@192.168.0.25
```

2. Check the health of the Ceph cluster:

```
$ sudo cephadm shell -- ceph health
```

   If the Ceph cluster has no issues, the command reports back **HEALTH_OK**. This means the Ceph cluster is safe to use.

3. Log in to an overcloud node that runs the Ceph monitor service and check the status of all OSDs in the Ceph cluster:

```
$ sudo cephadm shell -- ceph osd tree
```

4. Check the status of the Ceph Monitor quorum:

```
$ sudo cephadm shell -- ceph quorum_status
```

   This shows the monitors participating in the quorum and which one is the leader.

5. Verify that all Ceph OSDs are running:

```
$ sudo cephadm shell -- ceph osd stat
```

For more information on monitoring Ceph clusters, see Monitoring a Ceph Storage cluster in the *Red Hat Ceph Storage Administration Guide*.

11.4. MAPPING A BLOCK STORAGE (CINDER) TYPE TO YOUR NEW CEPH POOL

After you complete the configuration steps, make the performance tiers feature available to RHOSP tenants by using Block Storage (cinder) to create a type that is mapped to the **fastpool** tier that you created.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Source the **overcloudrc** file:

```
$ source overcloudrc
```

3. Check the Block Storage volume existing types:

```
$ cinder type-list
```

4. Create the new Block Storage volume fast_tier:

```
$ cinder type-create fast_tier
```

5. Check that the Block Storage type is created:

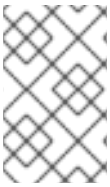
```
$ cinder type-list
```

6. When the **fast_tier** Block Storage type is available, set the **fastpool** as the Block Storage volume back end for the new tier that you created:

```
$ cinder type-key fast_tier set volume_backend_name=tripleo_ceph_fastpool
```

7. Use the new tier to create new volumes:

```
$ cinder create 1 --volume-type fast_tier --name fastdisk
```



NOTE

The Red Hat Ceph Storage documentation provides additional information and procedures for the ongoing maintenance and operation of the Ceph Storage cluster. See Product Documentation for Red Hat Ceph Storage 5 for this documentation.

CHAPTER 12. NATIVE CEPHFS POST-DEPLOYMENT CONFIGURATION AND VERIFICATION

You must complete some post-deployment configuration tasks before you create CephFS shares, grant user access, and mount CephFS shares.

- Map the Networking service (neutron) storage network to the isolated data center storage network.
- Make the storage provider network available to trusted tenants only through custom role based access control (RBAC). Do not share the storage provider network globally.
- Create a private share type.
- Grant access to specific trusted tenants.

After you complete these steps, the tenant compute instances can create, allow access to, and mount native CephFS shares.

Deploying native CephFS as a back end of the Shared File Systems service (manila) adds the following new elements to the overcloud environment:

- Storage provider network
- Ceph MDS service on the Controller nodes

The cloud administrator must verify the stability of the native CephFS environment before making it available to service users.

For more information about using the Shared File Systems service with native CephFS, see Configuring the Shared File Systems service (manila) in the *Storage Guide*.

12.1. CREATING THE STORAGE PROVIDER NETWORK

You must map the new isolated storage network to a Networking (neutron) provider network. The Compute VMs attach to the network to access native CephFS share export locations.

For information about network security with the Shared File Systems service (manila), see Hardening the Shared File Systems Service in *Hardening Red Hat OpenStack Platform*.

Procedure

The **openstack network create** command defines the configuration for the storage neutron network.

1. Source the overcloud credentials file:

```
$ source ~/<credentials_file>
```

   - Replace **<credentials_file>** with the name of your credentials file, for example, **overcloudrc**.

2. On an undercloud node, create the storage network:

```
(overcloud) [stack@undercloud-0 ~]$ openstack network create Storage --provider-network-type vlan --provider-physical-network datacentre --provider-segment 30
```

You can enter this command with the following options:

- For the **--provider-physical-network** option, use the default value **datacentre**, unless you set another tag for the br-isolated bridge through NeutronBridgeMappings in your tripleo-heat-templates.
- For the **--provider-segment** option, use the value set for the Storage isolated network in your network environment file. If this was not customized, the default environment file is **/usr/share/openstack-tripleo-heat-templates/network_data.yaml**. The VLAN associated with the Storage network value is **30** unless you modified the isolated network definitions.
- For the **--provider-network-type** option, use the value **vlan**.

12.2. CONFIGURING THE STORAGE PROVIDER NETWORK

Create a corresponding **StorageSubnet** on the neutron provider network. Ensure that the subnet is the same for the **storage_subnet** in the undercloud, and that the allocation range for the storage subnet and the corresponding undercloud subnet do not overlap.

Requirements

- The starting and ending IP range for the allocation pool
- The subnet IP range

Procedure

1. From an undercloud node, enter the following command:

```
[stack@undercloud ~]$ source ~/overcloudrc
```

2. Use the sample command to provision the network. Update the values to suit your environment.

```
(overcloud) [stack@undercloud-0 ~]$ openstack subnet create \
--allocation-pool start=172.17.3.10,end=172.17.3.149 \
--dhcp \
--network Storage \
--subnet-range 172.17.3.0/24 \
--gateway none StorageSubnet
```

- For the **--allocation-pool** option, replace the **start=172.17.3.10,end=172.17.3.149** IP values with the IP values for your network.
- For the **--subnet-range** option, replace the **172.17.3.0/24** subnet range with the subnet range for your network.

12.3. CONFIGURING ROLE-BASED ACCESS CONTROL FOR THE STORAGE PROVIDER NETWORK

After you identify the trusted tenants or projects that can use the storage network, configure role-based access control (RBAC) rules for them through the Networking service (neutron).

Requirements

Names of the projects that need access to the storage network

Procedure

1. From an undercloud node, enter the following command:

```
[stack@undercloud ~]$ source ~/overcloudrc
```

2. Identify the projects that require access:

```
(overcloud) [stack@undercloud-0 ~]$ openstack project list
+----------------------------------+---------+
| ID                               | Name    |
+----------------------------------+---------+
| 06f1068f79d2400b88d1c2c33eacea87 | demo    |
| 5038dde12dfb44fdaa0b3ee4bfe487ce | service |
| 820e2d9c956644c2b1530b514127fd0d | admin   |
+----------------------------------+---------+
```

3. Create network RBAC rules with the desired projects:

```
(overcloud) [stack@undercloud-0 ~]$ openstack network rbac create \
--action access_as_shared Storage \
--type network \
--target-project demo
```

Repeat this step for all of the projects that require access to the storage network.

12.4. CONFIGURING A DEFAULT SHARE TYPE

You can use the Shared File Systems service (manila) to define share types for the creation of shares with specific settings. Share types work like Block Storage volume types. Each type has associated settings, for example, extra specifications. When you invoke the type during share creation, the settings apply to the shared file system.

To secure the native CephFS back end against untrusted users, do not create a default share type. When a default share type does not exist, users are forced to specify a share type, and trusted users can use a custom private share type to which they have exclusive access rights.

If you must create a default share type for untrusted tenants, you can steer provisioning away from the native CephFS back end.

Procedure

1. Source the overcloud credentials file:

```
$ source ~/<credentials_file>
```

   - Replace **<credentials_file>** with the name of your credentials file, for example, **overcloudrc**.

2. Set an extra specification on the share type:

```
(overcloud) [stack@undercloud-0 ~]$ manila type-create default false
(overcloud) [stack@undercloud-0 ~]$ manila type-key default set share_backend_name='s!=
cephfs'
```

3. Create a private share type and provide trusted tenants with access to this share type:

```
(overcloud) [stack@undercloud-0 ~]$ manila type-create --is-public false nativecephfstype
false
(overcloud) [stack@undercloud-0 ~]$ manila type-key nativecephfstype set
share_backend_name='cephfs'
(overcloud) [stack@undercloud-0 ~]$ manila type-access-add nativecephfstype
<trusted_tenant_project_id>
```

- Replace **<trusted_tenant_project_id>** with the ID of the trusted tenant.

For more information about share types, see Creating share types in the *Storage Guide*.

12.5. VERIFYING CREATION OF ISOLATED STORAGE NETWORK

The **network_data.yaml** file used to deploy native CephFS as a Shared File Systems service back end creates the storage VLAN. Use this procedure to confirm you successfully created the storage VLAN.

Procedure

1. Log in to one of the Controller nodes in the overcloud.
2. Check the connected networks and verify the existence of the VLAN as set in the **network_data.yaml** file:

```
$ ip a
8: vlan30: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/ether 52:9c:82:7a:d4:75 brd ff:ff:ff:ff:ff:ff
    inet 172.17.3.144/24 brd 172.17.3.255 scope global vlan30
       valid_lft forever preferred_lft forever
    inet6 fe80::509c:82ff:fe7a:d475/64 scope link
       valid_lft forever preferred_lft forever
```

12.6. VERIFYING CEPH MDS SERVICE

Use the **systemctl status** command to verify the Ceph MDS service status.

Procedure

- Enter the following command on all Controller nodes to check the status of the MDS container:

```
$ systemctl status ceph-mds<@CONTROLLER-HOST>
```

Example:

```
$ systemctl status ceph-mds@controller-0.service

ceph-mds@controller-0.service - Ceph MDS
```

```
   Loaded: loaded (/etc/systemd/system/ceph-mds@.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2018-09-18 20:11:53 UTC; 6 days ago
 Main PID: 65066 (conmon)
   Tasks: 16 (limit: 204320)
   Memory: 38.2M
   CGroup: /system.slice/system-ceph\x2dmds.slice/ceph-mds@controller-0.service
           └─60921 /usr/bin/podman run --rm --net=host --memory=32000m --cpus=4 -v
/var/lib/ceph:/var/lib/ceph:z -v /etc/ceph:/etc/ceph:z -v
/var/run/ceph:/var/run/ceph:z -v /etc/localtime:/etc/localtime:ro>
```

12.7. VERIFYING CEPH CLUSTER STATUS

Verify the Ceph cluster status to confirm that the cluster is active.

Procedure

1. Log in to any Controller node.
2. From the Ceph monitor daemon, enter the following command:

```
$ sudo podman exec ceph-mon-controller-0 ceph -s
  cluster:
    id:     670dc288-cd36-4772-a4fc-47287f8e2ebf
    health: HEALTH_OK

  services:
    mon: 3 daemons, quorum controller-1,controller-2,controller-0 (age 14h)
    mgr: controller-1(active, since 8w), standbys: controller-0, controller-2
    mds: cephfs:1 {0=controller-2=up:active} 2 up:standby
    osd: 15 osds: 15 up (since 8w), 15 in (since 8w)

  task status:
    scrub status:
        mds.controller-2: idle

  data:
    pools: 6 pools, 192 pgs
    objects: 309 objects, 1.6 GiB
    usage: 21 GiB used, 144 GiB / 165 GiB avail
    pgs: 192 active+clean
```

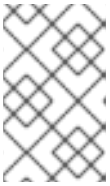
NOTE

There is one active MDS and two MDSs on standby.

3. To see a detailed status of the Ceph File System, enter the following command:

```
$ sudo ceph fs ls

name: cephfs metadata pool: manila_metadata, data pools: [manila_data]
```



NOTE

In this example output, **cephfs** is the name of Ceph File System that director creates to host CephFS shares that users create through the Shared File Systems service.

12.8. VERIFYING MANILA-SHARE SERVICE STATUS

Verify the status of the manila-share service.

Procedure

1. From one of the Controller nodes, confirm that **openstack-manila-share** started:

```
$ sudo pcs status resources | grep manila

* Container bundle: openstack-manila-share [cluster.common.tag/rhosp16-openstack-manila-share:pcmklatest]:
* openstack-manila-share-podman-0 (ocf::heartbeat:podman): Started controller-0
```

12.9. VERIFYING MANILA-API SERVICES ACKNOWLEDGES SCHEDULER AND SHARE SERVICES

Complete the following steps to confirm that the **manila-api** service acknowledges the scheduler and share services.

Procedure

1. Log in to the undercloud.
2. Enter the following command:

```
$ source /home/stack/overcloudrc
```

3. Enter the following command to confirm **manila-scheduler** and **manila-share** are enabled:

```
$ manila service-list

| Id | Binary          | Host              | Zone | Status | State | Updated_at |

| 2 | manila-scheduler | hostgroup         | nova | enabled | up | 2018-08-08T04:15:03.000000 |
| 5 | manila-share | hostgroup@cephfs | nova | enabled | up | 2018-08-08T04:15:03.000000 |
```

CHAPTER 13. CEPHFS NFS POST-DEPLOYMENT CONFIGURATION AND VERIFICATION

You must complete two post-deployment configuration tasks before you create NFS shares, grant user access, and mount NFS shares.

- Map the Networking service (neutron) StorageNFS network to the isolated data center Storage NFS network. You can omit this option if you do not want to isolate NFS traffic to a separate network.
- Create the default share type.

After you complete these steps, the tenant compute instances can create, allow access to, and mount NFS shares.

When you deploy CephFS through NFS as a back end of the Shared File Systems service (manila), you add the following new elements to the overcloud environment:

- StorageNFS network
- Ceph MDS service on the controllers
- NFS-Ganesha service on the controllers

As the cloud administrator, you must verify the stability of the CephFS through NFS environment before you make it available to service users.

13.1. CREATING THE STORAGE PROVIDER NETWORK

You must map the new isolated StorageNFS network to a Networking (neutron) provider network. The Compute VMs attach to the network to access share export locations that are provided by the NFS-Ganesha gateway.

For information about network security with the Shared File Systems service (manila), see Hardening the Shared File Systems Service in *Hardening Red Hat OpenStack Platform*.

Procedure

The **openstack network create** command defines the configuration for the StorageNFS neutron network.

1. Source the overcloud credentials file:

```
$ source ~/<credentials_file>
```

   - Replace **<credentials_file>** with the name of your credentials file, for example, **overcloudrc**.

2. On an undercloud node, create the StorageNFS network:

```
(overcloud) [stack@undercloud-0 ~]$ openstack network create StorageNFS --share  --
provider-network-type vlan --provider-physical-network datacentre --provider-segment 70
```

   You can enter this command with the following options:

- For the **--provider-physical-network** option, use the default value **datacentre**, unless you set another tag for the br-isolated bridge through NeutronBridgeMappings in your tripleo-heat-templates.
- For the **--provider-segment** option, use the VLAN value set for the StorageNFS isolated network in the heat template, **/usr/share/openstack-tripleo-heat-templates/network_data_ganesha.yaml**. This value is 70, unless the deployer modified the isolated network definitions.
- For the **--provider-network-type** option, use the value **vlan**.

13.2. CONFIGURING THE SHARED PROVIDER STORAGENFS NETWORK

Create a corresponding **StorageNFSSubnet** on the neutron-shared provider network. Ensure that the subnet is the same as the **storage_nfs** network definition in the **network_data.yml** file and ensure that the allocation range for the **StorageNFS** subnet and the corresponding undercloud subnet do not overlap. No gateway is required because the **StorageNFS** subnet is dedicated to serving NFS shares.

Prerequisites

- The start and ending IP range for the allocation pool.
- The subnet IP range.

13.2.1. Configuring the shared provider StorageNFS IPv4 network

Create a corresponding **StorageNFSSubnet** on the neutron-shared IPv4 provider network.

Procedure

1. Log in to an overcloud node.
2. Source your overcloud credentials.
3. Use the example command to provision the network and make the following updates:
   a. Replace the **start=172.17.0.4,end=172.17.0.250** IP values with the IP values for your network.
   b. Replace the **172.17.0.0/20** subnet range with the subnet range for your network.

```
[stack@undercloud-0 ~]$ openstack subnet create --allocation-pool
start=172.17.0.4,end=172.17.0.250 \
--dhcp --network StorageNFS --subnet-range 172.17.0.0/20 \
--gateway none StorageNFSSubnet
```

13.2.2. Configuring the shared provider StorageNFS IPv6 network

Create a corresponding **StorageNFSSubnet** on the neutron-shared IPv6 provider network.

Procedure

1. Log in to an overcloud node.

2. Use the sample command to provision the network, updating values as needed.
    - Replace the **fd00:fd00:fd00:7000::/64** subnet range with the subnet range for your network.

```
[stack@undercloud-0 ~]$ openstack subnet create --ip-version 6 --dhcp --network StorageNFS --subnet-range fd00:fd00:fd00:7000::/64 --gateway none --ipv6-ra-mode dhcpv6-stateful --ipv6-address-mode dhcpv6-stateful StorageNFSSubnet -f yaml
```

13.3. CONFIGURING A DEFAULT SHARE TYPE

You can use the Shared File Systems service (manila) to define share types for the creation of shares with specific settings. Share types work like Block Storage volume types. Each type has associated settings, for example, extra specifications. When you invoke the type during share creation, the settings apply to the shared file system.

With Red Hat OpenStack Platform (RHOSP) director, you must create a default share type before you open the cloud for users to access.

Procedure

- Create a default share type for CephFS with NFS:

```
$ manila type-create default false
```

For more information about share types, see Creating share types in the *Storage Guide*.

13.4. VERIFYING CREATION OF ISOLATED STORAGENFS NETWORK

The **network_data_ganesha.yaml** file used to deploy CephFS through NFS as a Shared File Systems service back end creates the StorageNFS VLAN. Complete the following steps to verify the existence of the isolated StorageNFS network.

Procedure

1. Log in to one of the controllers in the overcloud.
2. Enter the following command to check the connected networks and verify the existence of the VLAN as set in **network_data_ganesha.yaml**:

```
$ ip a
15: vlan310: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group default qlen 1000
    link/ether 32:80:cf:0e:11:ca brd ff:ff:ff:ff:ff:ff
    inet 172.16.4.4/24 brd 172.16.4.255 scope global vlan310
       valid_lft forever preferred_lft forever
    inet 172.16.4.7/32 brd 172.16.4.255 scope global vlan310
       valid_lft forever preferred_lft forever
    inet6 fe80::3080:cfff:fe0e:11ca/64 scope link
       valid_lft forever preferred_lft forever
```

13.5. VERIFYING CEPH MDS SERVICE

Use the **systemctl status** command to verify the Ceph MDS service status.

Procedure

- Enter the following command on all Controller nodes to check the status of the MDS container:

```
$ systemctl status ceph-mds<@CONTROLLER-HOST>
```

Example:

```
$ systemctl status ceph-mds@controller-0.service

ceph-mds@controller-0.service - Ceph MDS
   Loaded: loaded (/etc/systemd/system/ceph-mds@.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2018-09-18 20:11:53 UTC; 6 days ago
 Main PID: 65066 (conmon)
   Tasks: 16 (limit: 204320)
   Memory: 38.2M
   CGroup: /system.slice/system-ceph\x2dmds.slice/ceph-mds@controller-0.service
           └─60921 /usr/bin/podman run --rm --net=host --memory=32000m --cpus=4 -v
/var/lib/ceph:/var/lib/ceph:z -v /etc/ceph:/etc/ceph:z -v
/var/run/ceph:/var/run/ceph:z -v /etc/localtime:/etc/localtime:ro>
```

13.6. VERIFYING CEPH CLUSTER STATUS

Complete the following steps to verify Ceph cluster status.

Procedure

1. Log in to the active Controller node.
2. Enter the following command:

```
$ sudo ceph -s

cluster:
    id: 3369e280-7578-11e8-8ef3-801844eeec7c
    health: HEALTH_OK

  services:
    mon: 3 daemons, quorum overcloud-controller-1,overcloud-controller-2,overcloud-
controller-0
    mgr: overcloud-controller-1(active), standbys: overcloud-controller-2, overcloud-controller-0
    mds: cephfs-1/1/1 up  {0=overcloud-controller-0=up:active}, 2 up:standby
    osd: 6 osds: 6 up, 6 in
```

There is one active MDS and two MDSs on standby.

3. To check the status of the Ceph file system in more detail, enter the following command and replace **<cephfs>** with the name of the Ceph file system:

```
$ sudo ceph fs ls

name: cephfs, metadata pool: manila_metadata, data pools: [manila_data]
```

13.7. VERIFYING NFS-GANESHA AND MANILA-SHARE SERVICE STATUS

Complete the following step to verify the status of NFS-Ganesha and manila-share service.

Procedure

1. Enter the following command from one of the Controller nodes to confirm that **ceph-nfs** and **openstack-manila-share** started:

```
$ pcs status

ceph-nfs       (systemd:ceph-nfs@pacemaker):   Started overcloud-controller-1

podman container: openstack-manila-share [192.168.24.1:8787/rhosp-rhel8/openstack-manila-share:pcmklatest]
  openstack-manila-share-podman-0      (ocf::heartbeat:podman):        Started overcloud-controller-1
```

13.8. VERIFYING MANILA-API SERVICES ACKNOWLEDGES SCHEDULER AND SHARE SERVICES

Complete the following steps to confirm that the **manila-api** service acknowledges the scheduler and share services.

Procedure

1. Log in to the undercloud.
2. Enter the following command:

```
$ source /home/stack/overcloudrc
```

3. Enter the following command to confirm **manila-scheduler** and **manila-share** are enabled:

```
$ manila service-list

| Id | Binary          | Host              | Zone | Status | State | Updated_at |

| 2 | manila-scheduler | hostgroup         | nova | enabled | up | 2018-08-08T04:15:03.000000 |
| 5 | manila-share | hostgroup@cephfs | nova | enabled | up | 2018-08-08T04:15:03.000000 |
```

CHAPTER 14. REBOOTING THE ENVIRONMENT

It might become necessary to reboot the environment. For example, when you need to modify physical servers or recover from a power outage. In these types of situations, it is important to make sure your Ceph Storage nodes boot correctly.

You must boot the nodes in the following order:

1. **Boot all Ceph Monitor nodes first**- This ensures the Ceph Monitor service is active in your high availability Ceph cluster. By default, the Ceph Monitor service is installed on the Controller node. If the Ceph Monitor is separate from the Controller in a custom role, make sure this custom Ceph Monitor role is active.

2. **Boot all Ceph Storage nodes**- This ensures the Ceph OSD cluster can connect to the active Ceph Monitor cluster on the Controller nodes.

14.1. REBOOTING A CEPH STORAGE (OSD) CLUSTER

Complete the following steps to reboot a cluster of Ceph Storage (OSD) nodes.

Prerequisites

- On a Ceph Monitor or Controller node that is running the **ceph-mon** service, check that the Red Hat Ceph Storage cluster status is healthy and the pg status is **active+clean**:

  ```
  $ sudo cephadm -- shell ceph status
  ```

  If the Ceph cluster is healthy, it returns a status of **HEALTH_OK**.

  If the Ceph cluster status is unhealthy, it returns a status of **HEALTH_WARN** or **HEALTH_ERR**. For troubleshooting guidance, see the Red Hat Ceph Storage 5 Troubleshooting Guide .

Procedure

1. Log in to a Ceph Monitor or Controller node that is running the **ceph-mon** service, and disable Ceph Storage cluster rebalancing temporarily:

   ```
   $ sudo cephadm shell -- ceph osd set noout
   $ sudo cephadm shell -- ceph osd set norebalance
   ```

   NOTE

   If you have a multistack or distributed compute node (DCN) architecture, you must specify the Ceph cluster name when you set the **noout** and **norebalance** flags. For example: **sudo cephadm shell -c /etc/ceph/<cluster>.conf -k /etc/ceph/<cluster>.client.keyring**.

2. Select the first Ceph Storage node that you want to reboot and log in to the node.

3. Reboot the node:

   ```
   $ sudo reboot
   ```

4. Wait until the node boots.
5. Log in to the node and check the Ceph cluster status:

```
$ sudo cephadm -- shell ceph status
```

   Check that the **pgmap** reports all **pgs** as normal (**active+clean**).

6. Log out of the node, reboot the next node, and check its status. Repeat this process until you have rebooted all Ceph Storage nodes.
7. When complete, log in to a Ceph Monitor or Controller node that is running the **ceph-mon** service and enable Ceph cluster rebalancing:

```
$ sudo cephadm shell -- ceph osd unset noout
$ sudo cephadm shell -- ceph osd unset norebalance
```

NOTE

If you have a multistack or distributed compute node (DCN) architecture, you must specify the Ceph cluster name when you unset the **noout** and **norebalance** flags. For example: **sudo cephadm shell -c /etc/ceph/<cluster>.conf -k /etc/ceph/<cluster>.client.keyring**

8. Perform a final status check to verify that the cluster reports **HEALTH_OK**:

```
$ sudo cephadm shell ceph status
```

14.2. REBOOTING CEPH STORAGE OSDS TO ENABLE CONNECTIVITY TO THE CEPH MONITOR SERVICE

If a situation occurs where all overcloud nodes boot at the same time, the Ceph OSD services might not start correctly on the Ceph Storage nodes. In this situation, reboot the Ceph Storage OSDs so they can connect to the Ceph Monitor service.

Procedure

- Verify a **HEALTH_OK** status of the Ceph Storage node cluster:

```
$ sudo ceph status
```

CHAPTER 15. SCALING THE CEPH STORAGE CLUSTER

You can scale the size of your Ceph Storage cluster by adding or removing storage nodes.

15.1. SCALING UP THE CEPH STORAGE CLUSTER

As capacity and performance requirements change, you can scale up your Ceph Storage cluster to meet increased demands. Before doing so, ensure that you have enough nodes for the updated deployment. Then you can register and tag the new nodes in your Red Hat OpenStack Platform (RHOSP) environment.

To register new Ceph Storage nodes with director, complete this procedure.

Procedure

1. Log in to the undercloud node as the **stack** user.
2. Modify the ~/**overcloud-baremetal-deploy.yaml** to add the CephStorage nodes to the deployment.
   The following example file represents an original deployment with three CephStorage nodes.

```
- name: CephStorage
  count: 3
  instances:
    - hostname: ceph-0
      name: ceph-0
    - hostname: ceph-1
      name: ceph-2
    - hostname: ceph-2
      name: ceph-2
```

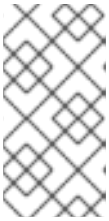
   The following example modifies this file to add three additional nodes.

```
- name: CephStorage
  count: 6
  instances:
    - hostname: ceph-0
      name: ceph-0
    - hostname: ceph-1
      name: ceph-2
    - hostname: ceph-2
      name: ceph-2
    - hostname: ceph-3
      name: ceph-3
    - hostname: ceph-4
      name: ceph-4
    - hostname: ceph-5
      name: ceph-5
```

3. Use the **openstack overcloud node provision** command with the updated ~/**overcloud-baremetal-deploy.yaml** file.

```
openstack overcloud node provision \
  --stack overcloud \
```

```
--network-config \
--output ~/overcloud-baremetal-deployed.yaml \
~/overcloud-baremetal-deploy.yaml
```



NOTE

This command will provision the configured nodes and and output an updated copy of ~/**overcloud-baremetal-deployed.yaml**. The new version updates the **CephStorage**. The **DeployedServerPortMap** and **HostnameMap** also contain the new storage nodes.

4. Use the **openstack overcloud deploy** command with the updated ~/**overcloud-baremetal-deployed.yaml** file.

```
openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm.yaml \
  -e deployed_ceph.yaml
  -e overcloud-baremetal-deploy.yaml
```

Result

The following actions occur when the **openstack overcloud deploy** command runs:

- The storage networks and firewall rules are configured on the new **CephStorage** nodes.
- The **ceph-admin** user is created on the new **CephStorage** nodes.
- The **ceph-admin** user public SSH key is distributed to the new **CephStorage** nodes so that **cephadm** can use SSH to add extra nodes.
- If a new **CephMon** or **CephMgr** node is added, the **ceph-admin** private SSH key is also distributed to that node.
- An updated Ceph specification is generated and installed on the bootstrap node. This updated specification will typically be available in /**home**/**ceph-admin**/**specs**/**ceph_spec.yaml** on the bootstrap node.
- The **cephadm** bootstrap process is skipped because **cephadm ls** indicates the Ceph containers are already running.
- The updated Ceph specification is applied and **cephadm** schedules the new nodes to join the Ceph cluster.

15.2. SCALING DOWN AND REPLACING CEPH STORAGE NODES

In some cases, you might need to scale down your Ceph Storage cluster or replace a Ceph Storage node. In either situation, you must disable and rebalance the Ceph Storage nodes that you want to remove from the overcloud to prevent data loss.

PROCEDURE

Do not proceed with this procedure if the Ceph Storage cluster does not have the capacity to lose OSDs.

1. Log in to the overcloud Controller node as the **tripleo-admin** user.
2. Use the **sudo cephadm shell** command to start a Ceph shell.
3. Use the **ceph osd tree** command to identify OSDs to be removed by server. In the following example we want to identify the OSDs of **ceph-2** host.

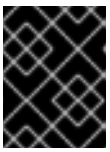
```
[ceph: root@oc0-controller-0 /]# ceph osd tree
ID  CLASS  WEIGHT   TYPE NAME          STATUS  REWEIGHT  PRI-AFF
-1         0.58557  root default
-7         0.19519  host ceph-2
 5    hdd  0.04880       osd.5          up      1.00000  1.00000
 7    hdd  0.04880       osd.7          up      1.00000  1.00000
 9    hdd  0.04880       osd.9          up      1.00000  1.00000
11    hdd  0.04880       osd.11         up      1.00000  1.00000
```

4. Export the Ceph cluster specification to a YAML file.

```
[ceph: root@oc0-controller-0 /]# ceph orch ls --export > spec.yml
```

5. Edit the exported specification file so that the applicable hosts are removed from the **service-type: osd hosts** list and the applicable hosts have the **placement: hosts** value removed.
6. Save the edited file.
7. Apply the modified Ceph specification file.

```
[ceph: root@oc0-controller-0 /]# ceph orch apply -i spec.yml
```



IMPORTANT

If you do not export and edit the Ceph specification file before removing the OSDs, the Ceph Manager will attempt to recreate the OSDs.

8. Use the command **ceph orch osd rm --zap <osd_list>** to remove the OSDs.

```
[ceph: root@oc0-controller-0 /]# ceph orch osd rm --zap 5 7 9 11
Scheduled OSD(s) for removal
[ceph: root@oc0-controller-0 /]# ceph orch osd rm status
OSD_ID HOST   STATE    PG_COUNT REPLACE  FORCE  DRAIN_STARTED_AT
7      ceph-2 draining 27       False    False  2021-04-23 21:35:51.215361
9      ceph-2 draining 8        False    False  2021-04-23 21:35:49.111500
11     ceph-2 draining 14       False    False  2021-04-23 21:35:50.243762
```

9. Use the command **ceph orch osd status** to check the status of OSD removal.

```
[ceph: root@oc0-controller-0 /]# ceph orch osd rm status
OSD_ID HOST   STATE    PG_COUNT REPLACE FORCE DRAIN_STARTED_AT
7      ceph-2 draining 34       False   False 2021-04-23 21:35:51.215361
11     ceph-2 draining 14       False   False 2021-04-23 21:35:50.243762
```

WARNING

Do not proceed with the next step until this command returns no results.

10. Use the command **ceph orch host drain <HOST>** to drain any remaining daemons.

```
[ceph: root@oc0-controller-0 /]# ceph orch host drain ceph-2
```

11. Use the command **ceph orch host rm <HOST>** to remove the host.

```
[ceph: root@oc0-controller-0 /]# ceph orch host rm ceph-2
```

12. End the Ceph shell session.
13. Log out of the **tripleo-admin** account.
14. Log in to the undercloud node as the **stack** user.
15. Modify the ~/**overcloud-baremetal-deploy.yaml** in the following ways:
    - Decrease the **count** attribute in the roles to be scaled down.
    - Add an **instances** entry for each node being unprovisioned. Each entry must contain the following:
        - The **name** of the baremetal node.
        - The **hostname** assigned to that node.
        - A **provisioned: false** value.
          The following example would remove the node **overcloud-compute-1**.

```
- name: Compute
  count: 1
  instances:
  - hostname: overcloud-compute-0
    name: node10
    # Removed from deployment due to disk failure
    provisioned: false
  - hostname: overcloud-compute-1
    name: node11
```

16. Use the **openstack overcloud node delete** command to remove the node.

```
openstack overcloud node delete \
--stack overcloud \
--baremetal-deployment ~/overcloud-baremetal-deploy.yaml
```

NOTE

A list of nodes to delete will be provided with a confirmation prompt before the nodes are deleted.

NOTE

If scaling down the Ceph cluster is temporary and the nodes removed will be restored later, the scaling up action can increment the **count** and set **provisioned: true** on nodes that were previously set **provisioned: false**. If the node will never reused, it can be set **provisioned: false** indefinitely and the scaling up action can specify a new instances entry.

+ The following file sample provides some examples of each instance.

+

```
- name: Compute
  count: 2
  instances:
  - hostname: overcloud-compute-0
    name: node10
    # Removed from deployment due to disk failure
    provisioned: false
  - hostname: overcloud-compute-1
    name: node11
  - hostname: overcloud-compute-2
    name: node12
```

CHAPTER 16. REPLACING A FAILED DISK

If a disk in your Ceph Storage cluster fails, you can replace it.

16.1. REPLACING A DISK

See Adding OSDs in the *Red Hat Ceph Storage Installation Guide* for information on replacing a failed disk.