Red Hat OpenStack Platform 16.0

CephFS via NFS Back End Guide for the Shared File System Service

Understanding, using, and managing the Shared File System Service with CephFS via NFS in OpenStack

Last Updated: 2020-04-09

Red Hat OpenStack Platform 16.0 CephFS via NFS Back End Guide for the Shared File System Service

Understanding, using, and managing the Shared File System Service with CephFS via NFS in OpenStack

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

This guide details the different procedures for installing, configuring, and verifying the Shared File System service (manila) with the Red Hat Ceph File System (CephFS) via NFS for the Red Hat OpenStack Platform environment.

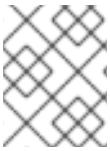
Table of Contents

PREFACE

Red Hat OpenStack Platform (Red Hat OpenStack Platform) provides the foundation to build a private or public Infrastructure-as-a-Service (IaaS) cloud on top of Red Hat Enterprise Linux. It offers a massively scalable, fault-tolerant platform for the development of cloud-enabled workloads.

The Shared File System service (manila) with Ceph File System (CephFS) via NFS enables cloud administrators to use the same Ceph cluster used for block and object storage to provide file shares via the NFS protocol. See the Shared File System service chapter in the *Storage Guide* for additional information.

This guide discusses concepts and procedures related to installing, configuring, deploying, and testing CephFS via NFS.



NOTE

For the complete suite of documentation for Red Hat OpenStack Platform, see Red Hat OpenStack Platform Documentation.

CHAPTER 1. OPENSTACK SHARED FILE SYSTEM SERVICE WITH CEPHFS VIA NFS

WARNING

The Red Hat OpenStack Platform (RHOSP) Shared File System service with CephFS via NFS is not available in RHOSP 16.0. The feature will be available with a future release of Red Hat Ceph Storage 4.0. Contact Red Hat Support to access the feature prior to the updated release. For more information, see BZ 1797075 and Known Issues in the *Release Notes*.

The OpenStack Shared File System service (manila) with Ceph File System (CephFS) via NFS provides a fault-tolerant NFS share service for the Red Hat OpenStack Platform. See the Shared File System service chapter in the *Storage Guide* for additional information.

1.1. INTRODUCTION TO CEPHFS VIA NFS

CephFS is the highly scalable, open-source distributed file system component of Ceph, a unified distributed storage platform. Ceph 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 System service enables users to create shares in CephFS and access them using NFS 4.1 via NFS-Ganesha. NFS-Ganesha controls access to the shares and exports them to clients via the NFS 4.1 protocol. The Shared File System service manages the life cycle of these shares from within OpenStack. When cloud administrators set up the service to use CephFS via NFS, these file shares come from the CephFS cluster, but are created and accessed as familiar NFS shares.

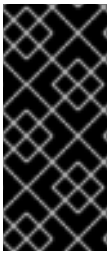
1.2. BENEFITS OF USING SHARED FILE SYSTEM SERVICE WITH CEPHFS VIA NFS

The Shared File System service (manila) with CephFS via NFS enables cloud administrators to use the same Ceph cluster they use for block and object storage to provide file shares through the familiar NFS protocol, which is available by default on most operating systems. CephFS maximizes Ceph clusters that are already used as storage back ends for other services in the OpenStack cloud, such as Block Storage (cinder), object storage, and so forth.

NOTE

Adding CephFS to an externally deployed Ceph cluster that was not configured by Red Hat OpenStack director is not supported at this time. Currently, only one CephFS back end can be defined in director.

This version of Red Hat OpenStack Platform fully supports the CephFS NFS driver (NFS-Ganesha), unlike the CephFS native driver, which is a Technology Preview feature.



IMPORTANT

Red Hat CephFS native driver is available only as a *Technology Preview*, and therefore is not fully supported by Red Hat.

For more information about Technology Preview features, see Scope of Coverage Details.

In CephFS via NFS deployments, the Ceph storage back end is separated from the user's network, which makes the underlying Ceph storage less vulnerable to malicious attacks and inadvertent mistakes.

Separate networks used for data-plane traffic and the API networks used to communicate with control plane services, such as Shared File System services, make file storage more secure.

The Ceph client is under administrative control. The end user controls an NFS client (an isolated user VM, for example) that has no direct access to the Ceph cluster storage back end.

1.3. CEPH FILE SYSTEM ARCHITECTURE

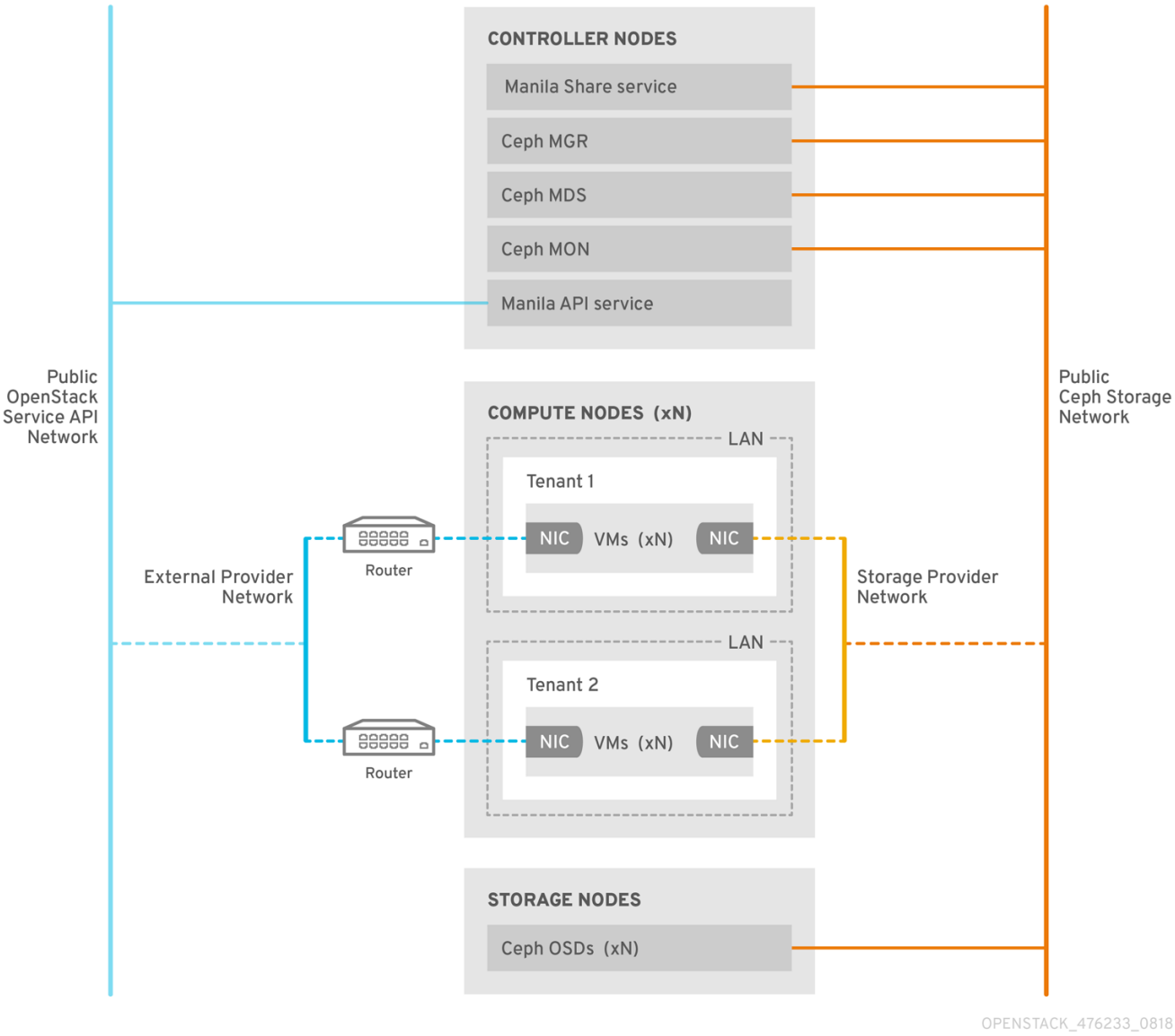
Ceph File System (CephFS) is a distributed file system that can be used with either NFS-Ganesha using the NFS v4 protocol (supported) or CephFS native driver (technology preview).

1.3.1. CephFS with native driver

The CephFS native driver combines the OpenStack Shared File System service (manila) and Red Hat Ceph Storage. When deployed via director, the controller nodes host the Ceph daemons, such as the manager, metadata servers (MDS), and monitors (MON) as well as the Shared File System services.

Compute nodes may host one or more tenants. Tenants, represented by the white boxes, which contain user-managed VMs (gray boxes with two NICs), access the **ceph** and **manila** daemons by connecting to them over the public Ceph storage network. This network also allows access to the data on the storage nodes provided by the Ceph Object Storage Daemons (OSDs). Instances (VMs) hosted on the tenant boot with two NICs: one dedicated to the storage provider network and the second to tenant-owned routers to the external provider network.

The storage provider network connects the VMs running on the tenants 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 tenant 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 properly.

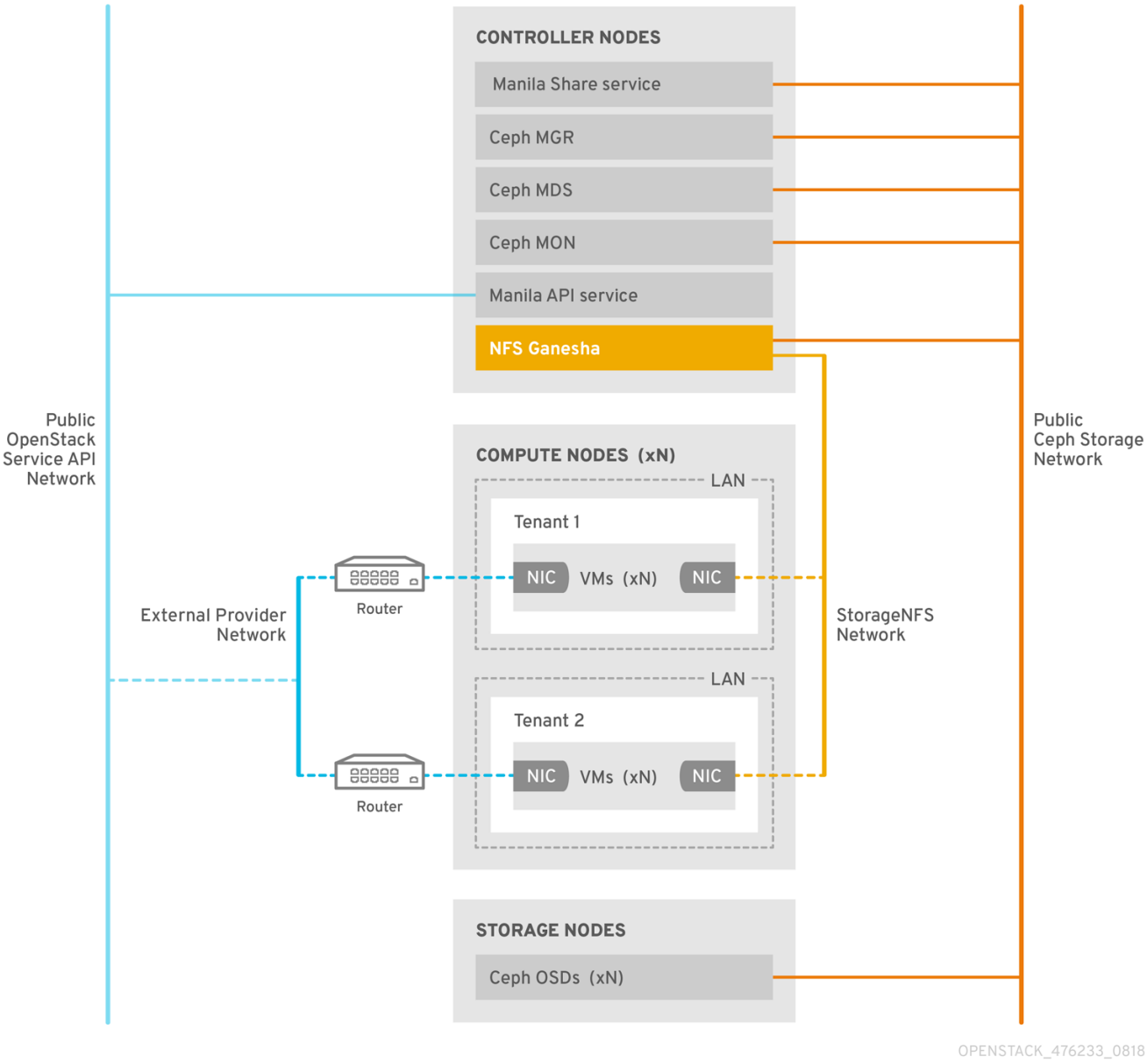


1.3.2. CephFS via NFS

The CephFS via NFS back end in the OpenStack Shared File Systems service (manila) is composed of Ceph metadata servers (MDS), the CephFS via NFS gateway (NFS-Ganesha), and the Ceph cluster service components. The Shared File System service's 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 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 along with the Ceph services.

Instances are booted with at least two NICs: one connects to the tenant router, and the second NIC connects to the StorageNFS network, which connects directly to the NFS-Ganesha gateway. The instance mounts shares using the NFS protocol. CephFS shares 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.



1.3.2.1. Ceph services and client access

In addition to the monitor, OSD, Rados Gateway (RGW), and manager services deployed when Ceph provides object and/or 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. OpenStack director's composable network feature is used to deploy this network and connect it to the controller nodes. The cloud administrator then configures the network as a 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, user VMs (nova instances) are provisioned with an additional NIC that connects to the Storage NFS network. Export locations for CephFS shares appear as standard NFS **IP:<path>** tuples using the NFS-Ganesha server's VIP on the StorageNFS network. Access control for user VMs is done using the user VM's IP on that network.

Neutron security groups prevent the user VM belonging to tenant 1 from accessing a user VM belonging to tenant 2 over the StorageNFS network. Tenants share the same CephFS filesystem but tenant data path separation is enforced because user VMs can only access files under export trees: /**path/to/share1/….**, /**path/to/share2/….**

1.3.2.2. Shared File System service with CephFS via NFS fault tolerance

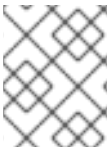
When OpenStack director starts the Ceph service daemons, they manage their own high availability (HA) state and, in general, there are multiple instances of these daemons running. By contrast, in this release, only one instance of NFS-Ganesha can serve file shares at a time.

To avoid a single point of failure in the data path for CephFS via NFS shares, NFS-Ganesha runs on an OpenStack 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 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 using the same virtual IP. Existing client mounts are preserved because they use the virtual IP for the export location of shares.

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

New connections, new lock-state, and so forth are refused until after a grace period of up to 90 seconds during which the server waits for clients to reclaim their locks. NFS-Ganesha keeps a list of the clients and will exit the grace period earlier, if it sees that all clients reclaimed their locks.



NOTE

The default value of the grace period is 90 seconds. This value is tunable through the NFSv4 **Grace_Period** configuration option.

CHAPTER 2. CEPHFS VIA NFS INSTALLATION

2.1. CEPHFS WITH NFS-GANESHA DEPLOYMENT

A typical Ceph file system (CephFS) via NFS installation in an OpenStack environment includes:

- OpenStack controller nodes running containerized Ceph metadata server (MDS), Ceph monitor (MON), manila, and NFS-Ganesha services. Some of these services may coexist on the same node or may 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 tenants to the NFS-Ganesha services for NFS share provisioning.

The Shared File System (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 (namely, **manila.share.drivers.cephfs.driver.CephFSDriver**) allows the Shared File System service to use CephFS as a back end. The Red Hat OpenStack Platform director configures the driver to deploy the NFS-Ganesha gateway so that the CephFS shares are presented via the NFS 4.1 protocol. In this document, this configuration is referred to as CephFS via NFS.

Using OpenStack director to deploy the Shared File System 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, refer to the Overcloud networks section of the *Director Installation and Usage Guide*.

While you can manually configure the Shared File System service by editing its node's **/etc/manila/manila.conf** file, any settings can be overwritten by the Red Hat OpenStack Platform director in future overcloud updates. The recommended method for configuring a Shared File System back end is through the director.

This section describes how to install CephFS via NFS in an integrated deployment managed by director.

NOTE

Adding CephFS to an externally deployed Ceph cluster that was not configured by Red Hat OpenStack director is not supported at this time. Currently, only one CephFS back end can be defined in director at a time.

2.1.1. Requirements

To use CephFS via NFS, you need a Red Hat OpenStack Platform version 13 or newer environment, which can be an existing or new OpenStack environment. CephFS works with Red Hat Ceph Storage version 3. See the *Deploying an Overcloud with Containerized Red Hat Ceph Guide* for instructions on how to deploy such an environment.

This document assumes that:

- The Shared File System service will be installed on controller nodes, as is the default behavior.
- The NFS-Ganesha gateway service will be installed on the controller's nodes Pacemaker cluster.

- Only a single instance of a CephFS back end will be used by the Shared File System Service. Other non-CephFS back ends can be used with the single CephFS back end.
- An extra network (StorageNFS) created by OpenStack Platform director used for the storage traffic.
- New Red Hat Ceph Storage version 3 cluster configured at the same time as CephFS via NFS.

2.1.2. File shares

File shares are handled slightly different between the OpenStack Shared File System service (manila), Ceph File System (CephFS), and Ceph via NFS.

The Shared File System service provides shares, where a share is an individual file system namespace and a unit of storage or sharing and a defined size (for example, subdirectories with quotas). Shared file system storage enables multiple clients because the file system is set up prior to when access is requested (versus block storage, which is set up at the time it is requested).

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

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

2.1.3. Isolated network used by CephFS via NFS

CephFS via NFS deployments use an extra isolated network, StorageNFS. This network is deployed so users can mount shares over NFS on that network without accessing the Storage or Storage Management networks which are reserved for infrastructure traffic.

For more information about isolating networks, see Basic network isolation in the *Advanced Overcloud Customization* guide.

2.2. INSTALLING OPENSTACK WITH CEPHFS VIA NFS AND A CUSTOM NETWORK_DATA FILE

Installing CephFS via NFS involves:

1. Installing the ceph-ansible package.
2. Preparing the overcloud container images with the **openstack overcloud image prepare** command.
3. Generating the custom roles file (**roles_data.yaml**) and **network_data.yaml** file.
4. Deploying Ceph, Shared File System service (manila), and CephFS using the **openstack overcloud deploy** command with custom roles and environments.
5. Configuring the isolated StorageNFS network and creating the default share type.

Examples use the standard **stack** user in the OpenStack environment.

Tasks should be performed in conjunction with an OpenStack installation or environment update.

2.2.1. Installing the ceph-ansible package

The OpenStack director requires the **ceph-ansible** package to be installed on an undercloud node to deploy containerized Ceph.

Procedure

1. Log in to an undercloud node.
2. Install the ceph-ansible package using **dnf install** with elevated privileges.

```
[stack@undercloud-0 ~]$ sudo dnf install -y ceph-ansible
[stack@undercloud-0 ~]$ sudo dnf list ceph-ansible
...
Installed Packages
ceph-ansible.noarch 3.1.0-0.1.el7
```

2.2.2. Preparing overcloud container images

Because all services are containerized in OpenStack, container images have to be prepared for the overcloud using the **openstack overcloud image prepare** command. Running this command with the additional options adds default images for the **ceph** and **manila** services to the container registry. Ceph MDS and NFS-Ganesha services use the same Ceph base container image.

For additional information on container images, refer to the Container Images for Additional Services section in the *Director Installation and Usage Guide*.

Procedure

1. From the undercloud, run the **openstack overcloud image prepare** command with **-e** to include these environment files:

```
$ openstack overcloud container image prepare \
  ...
  -e  /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-
ansible.yaml \
  -e  /usr/share/openstack-tripleo-heat-templates/environments/services/manila.yaml \
  ...
```

2. Use grep to verify the default images for the **ceph** and **manila** services are available in the **containers-default-parameters.yaml** file.

```
[stack@undercloud-0 ~]$ grep -E 'ceph|manila' composable_roles/docker-images.yaml
DockerCephDaemonImage: 192.168.24.1:8787/rhceph-beta/rhceph-4-rhel8:4-12
DockerManilaApiImage: 192.168.24.1:8787/rhosp-rhel8/openstack-manila-api:2019-01-16
DockerManilaConfigImage: 192.168.24.1:8787/rhosp-rhel8/openstack-manila-api:2019-01-16
DockerManilaSchedulerImage: 192.168.24.1:8787/rhosp-rhel8/openstack-manila-
scheduler:2019-01-16
DockerManilaShareImage: 192.168.24.1:8787/rhosp-rhel8/openstack-manila-share:2019-01-
16
```

2.2.2.1. Generating the custom roles file

The **ControllerStorageNFS** custom role is used to set up the isolated StorageNFS network. This role is similar to the default **Controller.yaml** role file with the addition of the StorageNFS network and the **CephNfs** service (indicated by **OS::TripleO::Services:CephNfs**).

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

For information about the **openstack overcloud roles generate** command, refer to the Roles section of the *Advanced Overcloud Customization Guide*.

Procedure

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

NOTE

If you have an existing **roles_data.yaml** file, modify it to add **ControllerStorageNfs**, **Compute**, and **CephStorage** services to the configuration file. Refer to the Roles section of the *Advanced Overcloud Customization Guide*.

1. Log in to an undercloud node.
2. Use the **openstack overcloud roles generate** command to 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
```

2.2.3. 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. These environments and roles are explained below.

Your overcloud deploy command will have the options below in addition to other required options.

Action	Option	Additional Information
Add the updated default containers from the **overcloud container image prepare** command	**-e /home/stack/containers-default-parameters.yaml**	Section 2.2.2, "Preparing overcloud container images"
Add the extra StorageNFS network with **network_data_ganesha.yaml**	**-n /usr/share/openstack-tripleo-heat-templates/network_data_ganesha.yaml**	Section 2.2.3.1, "StorageNFS and network_data_ganesha.yaml file"

Action	Option	Additional Information
Add the custom roles defined in **roles_data.yaml** file from the previous section	**-r /home/stack/roles_data.yaml**	Section 2.2.2.1, "Generating the custom roles file"
Deploy the Ceph daemons with **ceph-ansible.yaml**	**-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml**	Initiating Overcloud Deployment in the *Deploying an Overcloud with Containerized Red Hat Ceph* guide
Deploy the Ceph metadata server with **ceph-mds.yaml**	**-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-mds.yaml**	Initiating Overcloud Deployment in the *Deploying an Overcloud with Containerized Red Hat Ceph* guide
Deploy the **manila** service with the CephFS via NFS back end. Configures NFS-Ganesha via director.	**-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml**	Section 2.2.3.2, "**manila-cephfsganesha-config.yaml**"

The example below shows an **openstack overcloud deploy command** incorporating options to deploy CephFS via NFS-Ganesha, Ceph 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  /usr/share/openstack-tripleo-heat-templates/environments/net-single-nic-with-vlans.yaml  \
-e /home/stack/network-environment.yaml  \
-e/usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-mds.yaml  \
-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml
```

For additional information on the **openstack overcloud deploy** command, refer to the Deployment command section in the *Director Installation and Usage Guide*.

2.2.3.1. StorageNFS and network_data_ganesha.yaml file

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

The **network_data_ganesha.yaml** file contains an additional section that defines the isolated StorageNFS network. While the default settings will work for most installations, you will still need to edit the YAML file to add your network settings, including the VLAN ID, subnet, etc.

```
name: StorageNFS
```

```
enabled: true
vip: true
name_lower: storage_nfs
vlan: 70
ip_subnet: '172.16.4.0/24'
allocation_pools: [{'start': '172.16.4.4', 'end': '172.16.4.250'}]
ipv6_subnet: 'fd00:fd00:fd00:7000::/64'
ipv6_allocation_pools: [{'start': 'fd00:fd00:fd00:7000::10', 'end': 'fd00:fd00:fd00:7000:ffff:ffff:ffff:fffe'}]
```

For more information on composable networks, refer to the Using Composable Networks section in the *Advanced Overcloud Customization Guide*.

2.2.3.2. manila-cephfsganesha-config.yaml

The integrated environment file for defining a CephFS back end 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 System service. The back end default settings should work for most environments. The example shows the default values used by the director when deploying the Shared File System 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 ceph-ansible
  # but it's still managed by pacemaker
  OS::TripleO::Services::CephNfs: ../deployment/ceph-ansible/ceph-nfs.yaml

parameter_defaults:
  ManilaCephFSBackendName: cephfs (1)
  ManilaCephFSDriverHandlesShareServers: false (2)
  ManilaCephFSCephFSAuthId: 'manila' (3)
  ManilaCephFSCephFSEnableSnapshots: false (4)
  # 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. 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 will not handle the lifecycle. This is the only supported option.
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. The **false** value indicates that snapshots are not enabled. This feature is currently not supported.

For more information about environment files, refer to the Environment Files section in the *Director Installation and Usage Guide*.

2.2.4. Completing post-deployment configuration

Two post-deployment configuration items need to be completed prior to allowing users access:

- The neutron StorageNFS network must be mapped to the isolated data center Storage NFS network, and
- The default share type must be created.

Once these steps are completed, the tenant compute instances can create, allow access to, and mount NFS shares.

2.2.4.1. Configuring the isolated network

The new isolated StorageNFS network must be mapped to a neutron-shared provider network. The Compute VMs attach to this neutron network to access share export locations provided by the NFS-Ganesha gateway.

For general information about network security with the Shared File System service, see Hardening the Shared File System Service in the *Security and Hardening Guide*.

The **openstack network create** command defines the configuration for the StorageNFS neutron network. You can run this command with the following options:

- For **--provider-physical-network**, use the default value **datacentre**, unless you set another tag for the br-isolated bridge through NeutronBridgeMappings in your tripleo-heat-templates.
- For **--provider-segment**, 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 **--provider-network-type**, use the value **vlan**.

Procedure

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

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

2. On an undercloud node, run the **openstack network create** command to create the StorageNFS network:

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

2.2.4.2. Configure the shared provider StorageNFS network

Create a corresponding StorageNFSSubnet on the neutron shared provider network. Ensure that the subnet is the same for the storage_nfs_subnet in the undercloud, but make sure 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.

Requirements

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

2.2.4.2.1. Configuring the shared provider StorageNFS IPv4 network

Procedure

1. Log in to an overcloud node.
2. Use the sample command to provision the network, updating values as needed.
   a. Replace the **start=172.16.4.150,end=172.16.4.250** IP values with the IP values for your network.
   b. Replace the **172.16.4.0/24** subnet range with the subnet range for your network.

```
[stack@undercloud-0 ~]$ openstack subnet create --allocation-pool
start=172.16.4.150,end=172.16.4.250 --dhcp --network StorageNFS --subnet-range 172.16.4.0/24 --
gateway none StorageNFSSubnet
```

2.2.4.2.2. Configuring the shared provider StorageNFS IPv6 network

This feature is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. It should only be used for testing, and should not be deployed in a production environment. For more information about Technology Preview features, see Scope of Coverage Details.

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
```

2.2.4.3. Setting up a default share type

The Shared File System service allows you to define share types that you can use to create shares with specific settings. Share types work like Block Storage volume types: each type has associated settings (namely, extra specifications), and invoking the type during share creation applies those settings.

The OpenStack director expects a default share type. This default share type has to be created prior to opening the cloud for users to access. For CephFS with NFS, use the **manila type-create** command:

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

For information about share types, refer to the section Create and Manage Share Types of the *Storage Guide*.

CHAPTER 3. VERIFYING SUCCESSFUL CEPHFS VIA NFS DEPLOYMENT

Deploying CephFS via NFS as a back end of OpenStack Shared File System service (manila) adds new elements to the overcloud environment.

The new overcloud elements are:

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

See the Shared File System service chapter in the *Storage Guide* for additional information about using the Shared File System service with CephFS via NFS.

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

Prerequisites

- Completing the steps in Chapter 2, *CephFS via NFS Installation*

3.1. VERIFYING CREATION OF ISOLATED STORAGENFS NETWORK

The **network_data_ganesha.yaml** file used to deploy CephFS via NFS as a Shared File Services system back end creates the StorageNFS VLAN:

```
- name: StorageNFS
  enabled: true
  vip: true
  name_lower: storage_nfs
  vlan: 310
  ip_subnet: '172.16.4.0/24'
  allocation_pools: [{'start': '172.16.4.4', 'end': '172.16.4.250'}]
  ipv6_subnet: 'fd00:fd00:fd00:7000::/64'
  IPv6_allocation_pools: [{'start': 'fd00:fd00:fd00:7000::10', 'end': 'fd00:fd00:fd00:7000:ffff:ffff:ffff:fffe'}]
```

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. Run 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
```

3.2. VERIFYING CEPH MDS SERVICE

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

Procedure

1. Run the following command on all controllers to check the status of the MDS container:

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

Example:

```
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)
```

3.3. VERIFYING CEPH CLUSTER STATUS

Complete the following steps to verify Ceph cluster status.

Procedure

1. Log in to the active controller.
2. Run 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
```

NOTE

Notice there is one active MDS and two MDSs on standby.

3. To check the status of the Ceph file system in more detail, run the following command where **<cephfs>** is the name of the Ceph file system:

```
$ sudo ceph fs ls

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

3.4. 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. Run the following command from one of the controllers 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
```

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

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

Procedure

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

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

3. Run 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 | down | None  |
```