Managing Cisco NFVI Storage

This chapter describes basic architectural concepts that will help you understand the Cisco NFVI data storage architecture and data flow. It also provides techniques you can use to monitor the storage cluster health and the health of all systems that depend on it

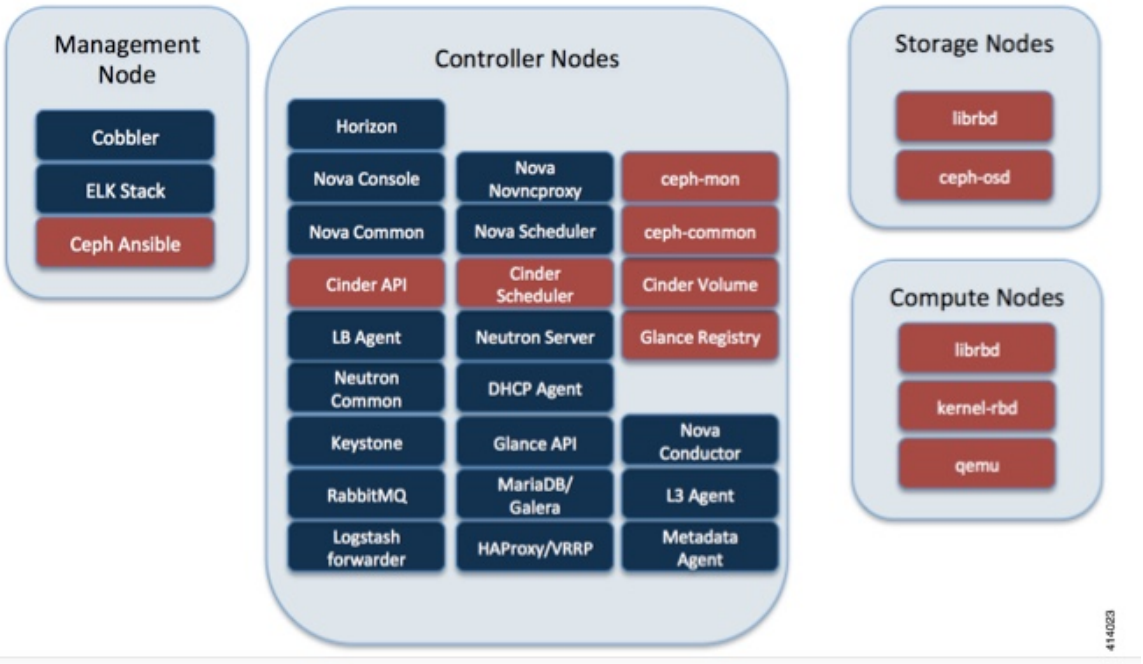
Cisco NFVI Storage Architecture

OpenStack has multiple storage back ends. Cisco NFVI uses the Ceph back end. Ceph supports both block and object storage and is therefore used to store VM images and volumes that can be attached to VMs. Multiple OpenStack services that depend on the storage backend include:

- Glance (OpenStack image service)—Uses Ceph to store images.
- Cinder (OpenStack storage service)—Uses Ceph to create volumes that can be attached to VMs.
- Nova (OpenStack compute service)—Uses Ceph to connect to the volumes created by Cinder.

The following figure shows the Cisco NFVI storage architecture component model.

Figure 1: Cisco NFVI Storage Architecture



Verifying and Displaying Ceph Storage Pools

Ceph is configured with four independent pools: images, volumes, vms, and backups. (A default rbd pool is used internally.) Each Ceph pool is mapped to an OpenStack service. The Glance service stores data in the images pool, and the Cinder service stores data in the volumes pool. The Nova service can use the vms pool to boot ephemeral disks directly from the Ceph cluster depending on how the NOVA_BOOT_FROM option in the ~/openstack-configs/setup_data.yaml was configured prior to Cisco NFVI installation. If NOVA_BOOT_FROM is set to ceph before you run the Cisco NFVI installation, the Nova service boot up from the Ceph vms pool. By default, NOVA_BOOT_FROM is set to local, which means that all VM ephemeral disks are stored as files in the compute nodes. Changing this option after installation does not affect the use of the vms pool for ephemeral disks.

The Glance, Cinder, and Nova OpenStack services depend on the Ceph cluster for backend storage. Therefore, they need IP connectivity to the controller nodes. The default port used to connect Glance, Cinder, and Nova to the Ceph cluster is 6789. Authentication through cephx is required, which means authentication tokens, called keyrings, must be deployed to the OpenStack components for authentication.

To verify and display the Cisco NFVI Ceph storage pools:

Step 1 Launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 2 Navigate to the Ceph Monitor container:

```
[root@controller_server-1 ~]# cephmon
```

Step 3 List the Ceph pools:

```
cephmon_4612 [root@controller_server-1 ~]# ceph osd lspools
0 rbd,1 images,2 volumes,3 vms,4 backups,
```

Step 4 List the images pool content:

```
cephmon_4612 [ceph@controller_server-1 /]$ rbd list images
a4963d51-d3b7-4b17-bf1e-2ebac07e1593
```

Checking the Storage Cluster Health

Cisco recommends that you perform a few verifications to determine whether the Ceph cluster is healthy and is connected to the Glance, Cinder, and Nova OpenStack services, which have Ceph cluster dependencies. The first task to check the health of the cluster itself by completing the following steps:

Step 1 From the Cisco NFVI management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 2 Navigate to the Ceph Monitor container:

```
[root@controller_server-1 ~]# cephmon
```

Step 3 Check the Ceph cluster status:

```
cephmon_4612 [ceph@controller_server-1 ceph]$ ceph status
```

Sample response:

```
cluster dbc29438-d3e0-4e0c-852b-170aaf4bd935
     health HEALTH_OK
     monmap e1: 3 mons at {ceph-controller_server-1=20.0.0.7:6789/0,
    ceph-controller_server-2=20.0.0.6:6789/0,ceph-controller_server-3=20.0.0.5:6789/0}
            election epoch 8, quorum 0,1,2 ceph-controller_server-3,
    ceph-controller_server-2,ceph-controller_server-1
     osdmap e252: 25 osds: 25 up, 25 in
      pgmap v593: 1024 pgs, 5 pools, 406 MB data, 57 objects
            2341 MB used, 61525 GB / 61527 GB avail
                1024 active+clean
```

This example displays three monitors, all in good health, and 25 object storage devices (OSDs). All OSDs show as up and in the cluster.

Step 4 To see a full listing of all OSDs sorted by storage node, enter:

```
cephmon_4612 [ceph@controller_server-1 ceph]$ ceph osd tree
```

Sample response:

```
ID WEIGHT   TYPE NAME               UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 60.18979 root default
-2 18.96994     host controller_server-2
 1  2.70999         osd.1                up  1.00000          1.00000
 5  2.70999         osd.5                up  1.00000          1.00000
 6  2.70999         osd.6                up  1.00000          1.00000
```

```
11  2.70999         osd.11              up  1.00000          1.00000
12  2.70999         osd.12              up  1.00000          1.00000
17  2.70999         osd.17              up  1.00000          1.00000
20  2.70999         osd.20              up  1.00000          1.00000
-3 18.96994     host controller_server-1
 0  2.70999         osd.0               up  1.00000          1.00000
 4  2.70999         osd.4               up  1.00000          1.00000
 8  2.70999         osd.8               up  1.00000          1.00000
10  2.70999         osd.10              up  1.00000          1.00000
13  2.70999         osd.13              up  1.00000          1.00000
16  2.70999         osd.16              up  1.00000          1.00000
18  2.70999         osd.18              up  1.00000          1.00000
-4 18.96994     host controller_server-3
 2  2.70999         osd.2               up  1.00000          1.00000
 3  2.70999         osd.3               up  1.00000          1.00000
 7  2.70999         osd.7               up  1.00000          1.00000
 9  2.70999         osd.9               up  1.00000          1.00000
14  2.70999         osd.14              up  1.00000          1.00000
15  2.70999         osd.15              up  1.00000          1.00000
19  2.70999         osd.19              up  1.00000          1.00000
-5  3.27997     host controller_server-4
21  0.81999         osd.21              up  1.00000          1.00000
22  0.81999         osd.22              up  1.00000          1.00000
23  0.81999         osd.23              up  1.00000          1.00000
24  0.81999         osd.24              up  1.00000          1.00000
```

What to do next

After you verify the Ceph cluster is in good health, check that the individual OpenStack components have connectivity and their authentication tokens—keyrings—match the Ceph Monitor keyrings. The following procedures show how to check the connectivity and authentication between Ceph and Glance, Ceph and Cinder, and Ceph and Nova.

Checking Glance Connectivity

The Glance API container must be connected to the Cisco NFVI controller nodes. Complete the following steps to verify the Glance to controller node connectivity:

Step 1 From the management node, examine the IP addresses of controller node:

```
[root@management-server-cisco ~]# cat /root/openstack-configs/mercury_servers_info
```

Step 2 From the management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 3 Navigate to the Glance API container:

```
[root@controller_server-1 ~]# glanceapi
```

Step 4 Check the Glance API container connectivity to the storage IP address of the controller node different from the one entered in Step 2:

```
glanceapi_4612 [glance@controller_server-1 /]$ curl <storage_ip_of_another_controller>:6789
```

If the connection is successful, no message is displayed, but you need to do Ctrl+C to terminate the connection:

```
glanceapi_4612 [glance@controller_server-1 /]$ curl 7.0.0.16:6789
```

If the connection is not successful, you see a message like the following:

```
glanceapi_4612 [glance@controller_server-1 /]$ curl 7.0.0.16:6789 curl: (7)
Failed connect to controller_server-2:6789; Connection refused
```

The above message indicates that the Ceph monitor running on the target controller node controller_server-2 is not listening on the specified port or there is no route to it from the Glance API container.

Checking one controller node should be enough to ensure one connection path available for the Glance API. As Cisco NFVI controller nodes run as part of an HA cluster, you should run Step 3 above targeting all the controller nodes in the Cisco NFVI pod.

What to do next

After you verify the Glance API connectivity to all Cisco NFVI controller nodes, check the Glance keyring to ensure that it matches with the Ceph monitor keyring.

Verifying Glance and Ceph Monitor Keyrings

Complete the following steps to verify the Glance API keyring matches the Ceph Monitor keyring.

Step 1 Launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 2 Navigate to the Glance API container:

```
[root@controller_server-1 ~]# glanceapi
```

Step 3 Check the Glance keyring content, for example:

```
glanceapi_4612 [glance@controller_server-1 /]$ cat /etc/ceph/client.glance.keyring [client.glance]
key = AQA/pY1XBAnHMBAAeS+0Wmh9PLZe1XqkIW/p0A==
```

Step 4 On the management node, check the CEPH cluster UUID:

```
[root@management-server-cisco ~]# cat /root/openstack-configs/ceph/fetch/ceph_cluster_uuid.conf
0e96e7f2-8175-44b3-ac1a-4f62de12ab9e
```

Step 5 Display the Ceph Glance keyring content:

```
[root@management-server-cisco ~]# cat
/root/openstack-configs/ceph/fetch/0e96e7f2-8175-44b3-ac1a-4f62de12ab9e/etc/ceph/ceph.client.glance.keyring

[mon.]

key = AQA/pY1XBAnHMBAAeS+0Wmh9PLZe1XqkIW/p0A==
```

Verify whether the keyring matches the Glance API keyring displayed in Step 3.

What to do next

A final check to ensure that Ceph and Glance are connected is to actually import a Glance image using Horizon or the Glance CLI. After you import an image, compare the IDs seen by Glance and by Ceph. They should match, indicating Ceph is handling the backend for Glance.

Verifying Glance Image ID on Ceph

The following steps verify Ceph is properly handling new Glance images by checking that the image ID for a new Glance image is the same as the image ID displayed in Ceph.

Step 1 From the management node, load the OpenStack authentication variables:

```
[root@management-server-cisco ~]# source ~/openstack-configs/openrc
```

Step 2 Import any Glance image. In the example below, a RHEL 7.1 qcow2 image is used.

```
[root@management-server-cisco images]# openstack image create
 "rhel" --disk-format qcow2 --container-format bare --file rhel-guest-image-7.1-20150224.0.x86_64.qcow2
```

Step 3 List the Glance images:

```
[root@management-server-cisco images]# openstack image list | grep rhel
| a4963d51-d3b7-4b17-bf1e-2ebac07e1593 | rhel
```

Step 4 Navigate to the Ceph Monitor container:

```
[root@controller_server-1 ~]# cephmon
```

Step 5 Display the contents of the Ceph images pool:

```
cephmon_4612 [ceph@controller_server-1 ceph]$ rbd list images | grep
a4963d51-d3b7-4b17-bf1e-2ebac07e1593
a4963d51-d3b7-4b17-bf1e-2ebac07e1593
```

Step 6 Verify that the Glance image ID displayed in Step 3 matches with the image ID displayed by Ceph.

Checking Cinder Connectivity

The Cinder volume container must have connectivity to the Cisco NFVI controller nodes. Complete the following steps to verify Cinder volume has connectivity to the controller nodes:

Step 1 From the management node, examine the IP addresses of controller node:

```
[root@management-server-cisco ~]# cat /root/openstack-configs/mercury_servers_info
```

Step 2 From the management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 3 Navigate to the Cinder volume container:

```
[root@controller_server-1 ~]# cindervolume
```

Step 4 Check the Cinder volume container connectivity to the storage IP address of controller node different from the one entered in Step 1:

```
cindervolume_4612 [cinder@controller_server-1 /]$ curl 7.0.0.16:6789
```

If the connection is successful, no message is displayed, but you need to do a Ctrl-C to terminate the connection:

```
cindervolume_4612 [cinder@controller_server-1 /]$ curl 7.0.0.16:6789
```

If the connection is not successful, you see a message like the following:

```
cindervolume_4612 [cinder@controller_server-1 /]$ curl controller_server-2:6789
curl: (7) Failed connect to controller_server-2:6789; Connection refused
```

A message like the one above means the Ceph monitor running on the target controller node controller_server-2 is not listening on the specified port or there is no route to it from the Cinder volume container.

Checking one controller node should be enough to ensure one connection path is available for the Cinder volume. However, because Cisco NFVI controller nodes run as part of an HA cluster, repeat Step 3 targeting all the controller nodes in the Cisco NFVI pod.

What to do next

After you verify the Cinder volume connectivity to all Cisco NFVI controller nodes, check the Cinder keyring to ensure it matches the Ceph monitor keyring.

Verifying Cinder and Ceph Monitor Keyrings

Complete the following steps to verify the Cinder volume keyring matches the Ceph Monitor keyring.

Step 1 From the management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 2 Navigate to the Cinder volume container:

```
[root@controller_server-1 ~]# cindervolume
```

Step 3 Check the Cinder keyring content, for example:

```
cindervolume_4612 [cinder@controller_server-1 /]$ cat /etc/ceph/client.cinder.keyring
[client.cinder]
key = AQA/pY1XBAnHMBAAeS+0Wmh9PLZe1XqkIW/p0A==
```

Step 4 On management node, check the CEPH cluster UUID:

```
[root@management-server-cisco ~]# cat /root/openstack-configs/ceph/fetch/ceph_cluster_uuid.conf
0e96e7f2-8175-44b3-ac1a-4f62de12ab9e
```

Step 5 Display the Ceph Cinder keyring content:

```
[root@management-server-cisco ~]# cat
```

```
/root/openstack-configs/ceph/fetch/0e96e7f2-8175-44b3-ac1a-4f62de12ab9e/etc/ceph/ceph.client.cinder.keyring
[client.cinder]
key = AQA/pY1XBAnHMBAAeS+0Wmh9PLZe1XqkIW/p0A==
```

Verify whether the keyring matches with the Cinder volume keyring displayed in Step 3.

What to do next

As a final Ceph and Cinder connectivity verification, import a Cinder image using Horizon or the Cinder CLI. After you import the image, compare the IDs seen by Cinder and by Ceph. If the IDs match, it indicates that Ceph is handling the backend for Cinder.

Verifying Cinder Volume ID on Ceph

The following steps verify Ceph is properly handling new Cinder volumes by checking that the volume ID for a new Cinder volume is the same as the volume ID displayed in Ceph.

Step 1 From the management node, load the OpenStack authentication variables:

```
[root@management-server-cisco ~]# source ~/openstack-configs/openrc
```

Step 2 Create an empty volume:

```
[root@management-server-cisco ~]# openstack volume create --size 5 ciscovol1
```

The preceding command creates a new 5 GB Cinder volume named ciscovol1.

Step 3 List the Cinder volumes:

```
[[root@management-server-cisco ~]# openstack volume list
+--------------------------------------+-----------+-----------+------+-------------+
| ID                                   | Name      | Status    | Size | Attached to |
+--------------------------------------+-----------+-----------+------+-------------+
| 3017473b-6db3-4937-9cb2-bd0ba1bf079d | ciscovol1 | available |    5 |             |
+--------------------------------------+-----------+-----------+------+-------------+
```

Step 4 Navigate to the Ceph Monitor container:

```
[root@controller_server-1 ~]# cephmon
```

Step 5 Display the contents of the Ceph volumes pool:

```
cephmon_4612 [ceph@controller_server-1 ceph]$ rbd list volumes
volume-3017473b-6db3-4937-9cb2-bd0ba1bf079d
```

Step 6 Verify that the Cinder volume ID displayed in Step 3 matches with the volume ID displayed by Ceph, excluding the "volume-" prefix.

Checking Nova Connectivity

The Nova libvirt container must have connectivity to the Cisco NFVI controller nodes. Complete the following steps to verify Nova has connectivity to the controller nodes:

Step 1 From the management node, examine the IP addresses of controller node:

```
[root@management-server-cisco ~]# cat /root/openstack-configs/mercury_servers_info
```

Step 2 From the management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@Computenode_server-1
```

Step 3 Navigate to the Nova libvirt container:

```
[root@compute_server-1 ~]# libvirt
```

Step 4 Check the Nova libvirt container connectivity to the storage address of the controller node different from the one entered in Step 1:

```
novalibvirt_4612 [root@compute_server-1 /]$ curl 7.0.0.16:6789
```

If the connection is successful, no message is displayed, but you need to do a Ctrl-C to terminate the connection:

If the connection is not successful, a message is displayed as follows:

```
novalibvirt_4612 [root@compute_server-1 /]$ curl 7.0.0.16:6789
curl: (7) Failed connect to controller_server-1:6789; Connection refused
```

The above message indicates that the Ceph monitor running on the target controller node controller_server-1 is not listening on the specified port or there is no route to it from the Nova libvirt container.

Checking one controller node is sufficient to ensure that one connection path is available for the Nova libvirt. As Cisco NFVI controller nodes run as part of an HA cluster, you should run Step 3 above targeting all the controller nodes in the Cisco NFVI pod.

What to do next

After you verify the Nova libvirt connectivity to all Cisco NFVI controller nodes, check the Nova keyring to ensure it matches the Ceph monitor keyring.

Verifying the Nova and Ceph Monitor Keyrings

Complete the following steps to verify the Nova libvirt keyring matches the Ceph Monitor keyring.

Step 1 From the management node, launch a SSH session to a controller node, for example:

```
[root@management-server-cisco ~]# ssh root@compute_server-1
```

Step 2 Navigate to the Nova libvert container:

```
[root@compute_server-1 ~]# libvirt
```

Step 3 Extract the libvirt secret that contains the Nova libvirt keyring:

```
novalibvirt_4612 [root@compute_server-1 /]# virsh secret-list
UUID                           Usage …
------------------------------------------------------------------
b5769938-e09f-47cb-bdb6-25b15b557e84 ceph client.cinder
```

Step 4 Get the keyring from the libvert secret:

```
novalibvirt_4612 [root@controller_server-1 /]# virsh secret-get-value
b5769938-e09f-47cb-bdb6-25b15b557e84 AQBApY1XQCBBEBAAroXvmiwmlSMEyEoXKl/sQA==
```

Step 5 On management node, check the CEPH cluster UUID::

```
[root@management-server-cisco ~]# cat /root/openstack-configs/ceph/fetch/ceph_cluster_uuid.conf
0e96e7f2-8175-44b3-ac1a-4f62de12ab9e
```

Step 6 Display the Ceph Cinder keyring content:

```
[root@management-server-cisco ~]# cat
/root/openstack-configs/ceph/fetch/0e96e7f2-8175-44b3-ac1a-4f62de12ab9e/etc/ceph/ceph.client.cinder.keyring
[client.cinder]

key = AQBApY1XQCBBEBAAroXvmiwmlSMEyEoXKl/sQA==
```

Step 7 Verify whether the keyring matches with the Nova libvirt keyring displayed in Step 3.

Note In the above example, the Cinder keyring is checked even though this procedure is for the Nova libvirt keyring. This occurs because the Nova services need access to the Cinder volumes and so authentication to Ceph uses the Cinder keyring.

What to do next

Complete a final check to ensure that Ceph and Nova are connected by attaching a Nova volume using Horizon or the Nova CLI. After you attach the Nova volume, check the libvirt domain.

Working with Multi-Backend Ceph

The OpenStack component that provides an API to create block storage for cloud is called OpenStack Block Storage service or Cinder. Cinder requires you to configure single backend (by default) or multiple backend.

Cisco VIM supports the following Cinder backends either configured in isolation or parallel.

- Storage nodes full of SSDs disks
- Storage nodes full of HDDs disks

Choosing multi-backend ceph is currently a day-0 option, that is, the cloud administrator must choose single backend or multi-backend storage option at the beginning. It is recommended to have four nodes with a minimum being three nodes for each storage type (HDD or SSD).

To enable support for Cinder multi-backends, update the `setup_data.yaml` to include the `osd_disk_type` as HDD/SSD under 'hardware_info' of the storage server as given below:

```
storage-hdd-server-1:  ---□ Need mininmum of 3; recommend to have 4
         cimc_info: {cimc_ip: <cimc_ip}
```

```
   hardware_info: {osd_disk_type: HDD}
   rack_info: {rack_id: RackA}
storage-ssd-server-1:  --□ ---□ Need mininmum of 3; recommend to have 4
  cimc_info: {cimc_ip: <cimc_ip>}
  hardware_info: {osd_disk_type: SSD}
  rack_info: {rack_id: RackB}
```

After successful deployment of Cisco VIM, follow the below steps to create Cinder multi-backends:

Step 1 Log into Horizon of the cloud and navigate to <**Admin/Volume/Volume Types** tab

a) Specify the Name and Description (Optional). The Name can be **volume-ssd** or **volume-hdd** for SSD or HDD, respectively
b) Click the dropdown and select **view extra specs**
c) Click on **Create** and update the **Key** with **volume_backend_name**.
d) Enter the **Value** as given below:

For SSD, the Value is **ceph-ssd**

For HDD, the Value is **ceph**

Step 2 Create Volume from Horizon for each backend type

a) Choose **Project/Volumes/Volumes**
b) Click **Create Volume**

Volume name : SSD volume (example)

Description: Optional

Volume Source : use the default

Type : Choose the volume type from the dropdown. It can be volume-ssd/volume-hdd

Step 3 Attach the volume to Instance.

a) Navigate to **Project/compute/instance**.
b) Select the instance to be attached the volume.
c) Click on the drop down **Attach volume** and select the Volume ID from the dropdown.

Verifying Nova Instance ID

From the management node, complete the following steps to verify the Nova instance ID of a guest VM having a cinder volume attached::

Step 1 Load the OpenStack authentication variables:

```
[root@management-server-cisco installer]# source ~/openstack-configs/openrc
```

Step 2 List the Nova instances:

```
[root@management-server-cisco images]# nova list
+--------------------------------------+------------+--------+--------
| ID                                   | Name       | Status | Task
```

```
+--------------------------------------+------------+--------+--------
| 77ea3918-793b-4fa7-9961-10fbdc15c6e5 | cisco-vm   | ACTIVE | -
+--------------------------------------+-----------+-------+-
```

Step 3 Show the Nova instance ID for one of the instances:

```
[root@management-server-cisco images]# nova show
77ea3918-793b-4fa7-9961-10fbdc15c6e5 | grep instance_name
| OS-EXT-SRV-ATTR:instance_name        | instance-00000003
```

The Nova instance ID in this example is instance-00000003. This ID will be used later with the virsh command. Nova instance IDs are actually the libvirt IDs of the libvirt domain associated with the Nova instance.

Step 4 Identify the compute node where the VM was deployed:

```
[root@management-server-cisco images]# nova show 77ea3918-793b-4fa7-9961-10fbdc15c6e5 | grep
hypervisor
| OS-EXT-SRV-ATTR:hypervisor_hostname  | compute_server-1
```

The compute node in this case is compute_server-1. You will connect to this compute node to call the virsh commands. Next, you get the volume ID from the libvirt domain in the Nova libvirt container.

Step 5 Launch a SSH session to the identified compute node, compute_server-1:

```
[root@management-server-cisco ~]# ssh root@compute_server-1
```

Step 6 Navigate to the Nova libvirt container:

```
[root@compute_server-1 ~]# libvirt
```

Step 7 Get the instance libvirt domain volume ID:

```
novalibvirt_4612 [root@compute_server-1 /]# virsh dumpxml instance-00000003 | grep rbd
 <source protocol='rbd' name='volumes/volume-dd188a5d-f822-4769-8a57-c16694841a23'>
```

Step 8 Launch a SSH session to a controller node:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 9 Navigate to the Ceph Monitor container:

```
[root@compute_server-1 ~]# cephmon
```

Step 10 Verify volume ID matches the ID in Step 7:

```
cephmon_4612 [ceph@controller_server-1 ceph]
$ rbd list volumes | grep volume-dd188a5d-f822-4769-8a57-c16694841a23
volume-dd188a5d-f822-4769-8a57-c16694841a23
```

Displaying Docker Disk Space Usage

Docker supports multiple storage back ends such as Device Mapper, thin pool, overlay, and AUFS. Cisco VIM uses the devicemapper storage driver because it provides strong performance and thin provisioning. Device Mapper is a kernel-based framework that supports advanced volume management capability. Complete the following steps to display the disk space used by Docker containers.

Step 1 Launch a SSH session to a controller or compute node, for example:

```
[root@management-server-cisco ~]# ssh root@controller_server-1
```

Step 2 Enter the docker info command to display the disk space used by Docker containers:

```
[root@controller_server_1 ~]# docker info
Containers: 24
Images: 186
Storage Driver: devicemapper
 Pool Name: vg_var-docker--pool
 Pool Blocksize: 524.3 kB
 Backing Filesystem: xfs
 Data file:
 Metadata file:
 Data Space Used: 17.51 GB
 Data Space Total: 274.9 GB
 Data Space Available: 257.4 GB…
```

Reconfiguring SwiftStack Integration

Cisco VIM provides integration with SwiftStack, an object storage solution. The key aspect of the SwiftStack integration is to add a SwfitStack endpoint to an existing pod running on Cisco VIM through the reconfigure option. In this case the SwiftStack is installed and managed outside the Cisco VIM ahead of time, and the VIM orchestrator adds the relevant Keystone configuration details to access the SwiftStack endpoint (see the Cisco VIM install guide for more details of SwiftStack).

The following options support the SwiftStack reconfiguration:

- Enable SwiftStack integration if it is not present.
- Reconfigure the existing SwiftStack PAC endpoint to point to a different cluster (cluster_api_endpoint).
- Reconfigure the Reseller_prefix of the existing SwiftStack installation.
- Reconfigure the admin password (admin_password) of an existing SwiftStack Install.

Integrating SwiftStack over TLS

The automation supports SwiftStack integration over TLS. To enable TLS, the CA root certificate must be presented as part of the `/root/openstack-configs/haproxy-ca.crt` file. The protocol parameter within the SWIFTSTACK stanza must be set to https. As a pre-requisite, the SwiftStack cluster needs to be configured to enable HTTPS connections for the SwiftStack APIs with termination at the proxy servers.

The following section needs to be configured in the Setup_data.yaml file.

```
########################################
# Optional Swift configuration section
########################################
# SWIFTSTACK: # Identifies the objectstore provider by name
#   cluster_api_endpoint: <IP address of PAC (proxy-account-container) endpoint>
#   reseller_prefix: <Reseller_prefix as configured for Keysone Auth,AuthToken support in
Swiftstack E.g KEY_>
#   admin_user: <admin user for swift to authenticate in keystone>
#   admin_password: <swiftstack_admin_password>
#   admin_tenant: <The service tenant corresponding to the Account-Container used by
Swiftstack
    protocol: <http or https> # protocol that swiftstack is running on top
```

Note The operator should pay attention while updating the settings to ensure that SwiftStack over TLS are appropriately pre-configured in the customer-managed SwiftStack controller as specified in the Install guide.

To initiate the integration, copy the `setupdata` into a local directory by running the following command:

```
[root@mgmt1 ~]# cd /root/
[root@mgmt1 ~]# mkdir MyDir
[root@mgmt1 ~]# cd MyDir
[root@mgmt1 ~]# cp /root/openstack-configs/setup_data.yaml <my_setup_data.yaml>
```

Update the `setupdata` by running the following command:

```
[root@mgmt1 ~]# vi my_setup_data.yaml (update the setup_data to include SwiftStack info)
```

Run the reconfiguration command as follows:

```
[root@mgmt1 ~]# cd ~/installer-xxxx
[root@mgmt1 ~]# ciscovim reconfigure --setupfile ~/MyDir/<my_setup_data.yaml>
```

Cinder Volume Backup on SwiftStack

Cisco VIM enables cinder service to be configured to backup its block storage volumes to the SwiftStack object store. This feature is automatically configured if the SWIFTSTACK stanza is present in the setup_data.yaml file. The mechanism is to authenticate against SwiftStack during volume backups leverages. The same keystone SwiftStack endpoint is configured to manage objects. The default SwiftStack container that manages cinder volumes within the account (Keystone Tenant as specified by admin_tenant) is currently defaulted to volumebackups.

Reconfiguring Administrator Source Networks

To access the administrator services, Cisco VIM provides source IP based filtering of network requests on the management node. These services include SSH and Kibana dashboard access. When the services are configured all admin network requests made to the management node are dropped, except the white listed addresses in the configuration.

Reconfiguring administrator source network supports the following options:

- Set administrator source network list: Network addresses can be added or deleted from the configuration; the list is replaced in whole during a reconfigure operation.
- Remove administrator source network list: If the **admin_source_networks** option is removed, then the source address does not filter the incoming admin service requests.

The following section needs to be configured in the Setup_data.yaml file:

```
admin_source_networks: # optional, host based firewall to white list admin's source IP
  - 10.0.0.0/8
  - 172.16.0.0/12
```

Note The operator has to be careful while updating the source networks. If the list is misconfigured, operators may lock themselves out of access to the management node through SSH. If it is locked, an operator must log into the management node through the console port to repair the configuration.

To initiate the integration, copy the `setupdata` into a local directory by running the following command:

```
[root@mgmt1 ~]# cd /root/
[root@mgmt1 ~]# mkdir MyDir
[root@mgmt1 ~]# cd MyDir
[root@mgmt1 ~]# cp /root/openstack-configs/setup_data.yaml <my_setup_data.yaml>
```

Update the `setupdata` by running the following command:

```
[root@mgmt1 ~]# vi my_setup_data.yaml (update the setup_data to include SwiftStack info)
```

Run the reconfiguration command as follows:

```
[root@mgmt1 ~]# cd ~/installer-xxxx
[root@mgmt1 ~]# ciscovim reconfigure --setupfile ~/MyDir/<my_setup_data.yaml>
```

Password Reset for Cisco VIM Management Node

Run the following command to reset the Root Password of Cisco VIM management node **RHEL-7 / systemd** .

1. Boot your system and wait until the **GRUB2** menu appears.
2. In the **boot loader** menu, highlight any entry and press **e**.
3. Find the line beginning with linux. At the end of this line, append the following:

   ```
   init=/bin/sh
   ```

   Or if you face any alarm, instead of **ro** change **rw** to **sysroot** as shown in the following example:

   ```
   rw init=/sysroot/bin/sh
   ```

4. Press **Ctrl+X** to boot the system using the options you edited.

   Once the system boots, you can see the shell prompt without having to enter any user name or password:

   ```
   sh-4.2#
   ```

5. Load the installed SELinux policy by running the following command:

   ```
   sh-4.2# /usr/sbin/load_policy -i
   ```

6. Execute the following command to remount your root partition:

   ```
   sh4.2#
       mount -o remount,rw /
   ```

7. Reset the root password by running the following command:

   ```
   sh4.2# passwd root
   ```

When prompted, enter your new root password and click **Enter** key to confirm. Enter the password for the second time to make sure you typed it correctly and confirm with **Enter** again. If both the passwords match, a confirmation message appears.

8. Execute the following command to remount the root partition again, this time as read-only:

```
sh4.2#
    mount -o remount,ro /
```

9. Reboot the system. Now you can log in as the root user using the new password set up during this procedure.

   To reboot the system, enter **`exit`** and **`exit`** again to leave the environment and reboot the system.

References: https://access.redhat.com/solutions/918283.

Ceph Storage Expansion

From release Cisco VIM 3.0.0, a command `expand-storage` is available to add disks so as to expand an already deployed Ceph storage cluster. You can deploy the storage nodes in the Openstack PoD in one of two ways:

- Combination of HDD and SSD drives with Ceph deployed with dedicated journals.
- All SSD drives with Ceph deployed with collocated journals.

Note The expand-storage command is supported only on storage nodes with a combination of HDD and SSD drives.

You must install disk drives based on how the node was originally deployed. If you have a storage node with a 1 SSD/4 HDDs ratio, insert disks with that same SSD/HDD ratio for expanding the storage. The expand-storage command looks for blocks of disks with that ratio during the expansion process, and installs one block at a time.

Workflow

- Use ciscovim list-nodes to find a node with a role of “block-storage”
- Insert a block of disks based on your storage deployment into the target node
- Log into the CIMC of the target node and navigate to the storage panel.
- Verify that no disks are reporting errors.
- If any disk reports foreign configuration, clear the configuration.
- Enable JBOD if RAID controller is used.
- Run cloud-sanity and ensure that no failure occurs.
- Run osdmgmt check-osds to check the state and number of the OSDs currently installed.
- Run expand-storage with the name of the target storage node.
- Run osdmgmt check-osds to check the state and number of the OSDs .

- Compare the outputs of the two osdmgmt check-osd commands and verify whether the new disks were added as additional OSDs.

The expand storage command runs in the same manner as other ciscovim commands but with various steps executed at a time.The command execution is stopped in case of any failure. You can view the logs once the command execution is complete. The steps for the expand-storage command are:

- Hardware validations
- Baremetal
- CEPH for expansion
- VMTP

Command Help

```
$ ciscovim help expand-storage
usage: ciscovim expand-storage --setupfile SETUPFILE [-y] <node>

Expand storage node capacity

Positional arguments:
  <node>                Expand Storage capacity of a storage node

Optional arguments:
  --setupfile SETUPFILE  <setupdata_file>. Mandatory for any POD management
                         operation.
  -y, --yes              Yes option to perform the action
```

Workflow command examples:

To expand the storage of node i13-27-test, get the current number/state of the OSDs in the cluster.

```
$ ciscovim list-nodes
+-------------+--------+---------------+----------------+
|  Node Name  | Status |      Type     | Management IP |
+-------------+--------+---------------+----------------+
|    i13-20   | Active |    control    |    15.0.0.7   |
|    i13-21   | Active |    control    |    15.0.0.8   |
|    i13-22   | Active |    control    |    15.0.0.5   |
|    i13-23   | Active |    compute    |    15.0.0.6   |
|    i13-24   | Active |    compute    |   15.0.0.10   |
|    i13-25   | Active | block_storage |   15.0.0.11   |
|    i13-26   | Active | block_storage |    15.0.0.9   |
| i13-27-test | Active | block_storage |    15.0.0.4   |
+-------------+--------+---------------+----------------+

ciscovim osdmgmt show check-osds --id <id>
+--------------------+-------------+---------------+-----------+---------+
| Message            | Host        | Role          | Server    | State   |
+--------------------+-------------+---------------+-----------+---------+
| Overall OSD Status | i13-25      | block_storage | 15.0.0.11 | Optimal |
|                    | i13-26      | block_storage | 15.0.0.9  | Optimal |
|                    | i13-27-test | block_storage | 15.0.0.4  | Optimal |
|                    |             |               |           |         |
| Number of OSDs     | i13-25      | block_storage | 15.0.0.11 | 10      |
|                    | i13-26      | block_storage | 15.0.0.9  | 10      |
|                    | i13-27-test | block_storage | 15.0.0.4  | 12       |
+--------------------+-------------+---------------+-----------+---------+
+-------------+--------+--------+----+------------+-----------+-----------
```

```
| Host        | OSDs   | Status | ID | HDD Slot   | Path      | Mount
+-------------+--------+--------+----+------------+-----------+-----------
.
omitted for doc
.
| i13-27-test | osd.2  | up     | 2  |  4 (JBOD)  | /dev/sda1 |
|             | osd.5  | up     | 5  |  3 (JBOD)  | /dev/sdb1 |
|             | osd.8  | up     | 8  |  6 (JBOD)  | /dev/sdc1 |
|             | osd.11 | up     | 11 |  2 (JBOD)  | /dev/sdd1 |
|             | osd.14 | up     | 14 |  5 (JBOD)  | /dev/sde1 |
|             | osd.19 | up     | 19 |  9 (JBOD)  | /dev/sdi1 |
|             | osd.24 | up     | 24 |  10 (JBOD) | /dev/sdj1 |
|             | osd.27 | up     | 27 |  8 (JBOD)  | /dev/sdl1 |
|             | osd.28 | up     | 28 |  12 (JBOD) | /dev/sdm1 |
|             | osd.29 | up     | 29 |  11 (JBOD) | /dev/sdn1 |
|             | osd.30 | up     | 30 |  13 (JBOD) | /dev/sdo1 |
|             | osd.31 | up     | 31 |  17 (JBOD) | /dev/sdp1 |
+-------------+--------+--------+----+------------+-----------+
```

Run the expand-storage command

```
# ciscovim expand-storage i13-27-test --setupfile setup_data.yaml

Perform the action. Continue (Y/N)Y

Monitoring StorageMgmt Operation

. . . . Cisco VIM Runner logs
The logs for this run are available in
<ip>:/var/log/mercury/05f068de-86fd-479c-afda-c54b14ffdd3e


############################################
Cisco Virtualized Infrastructure Manager
############################################


[1/3][VALIDATION: INIT]                                                    [   /   ]
 0min 0sec
Management Node Validations!
.
.
Omitted for doc
.
.
[1/3][VALIDATION: Starting HW Validation, takes time!!!]         [ DONE! ]

Ended Installation [VALIDATION] [Success]

[2/3][CEPH: Checking for Storage Nodes]                          [ DONE! ]
[2/3][CEPH: Creating Ansible Inventory]                          [ DONE! ]
.
.
Omitted for doc
.
.
[2/3][CEPH: Waiting for server to come back first try]           [ DONE! ]

Ended Installation [CEPH] [Success]
VMTP Starts

/home/vmtp/.ssh/id_rsa already exists.
.
.
```

```
Omitted for doc
.
.

[3/3][VMTP: INIT]                                                [ DONE! ]

Ended Installation [VMTP] [Success]

The logs for this run are available in
<ip>:/var/log/mercury/05f068de-86fd-479c-afda-c54b14ffdd3e
===========

Check the OSDs
ciscovim osdmgmt create check-osds
+------------+--------------------------------------+
| Field      | Value                                |
+------------+--------------------------------------+
| action     | check-osds                           |
| command    | create                               |
| created_at | 2019-01-07T19:00:23.575530+00:00     |
| id         | adb56a08-fdc5-4810-ac50-4ea6c6b38e3f |
| locator    | False                                |
| osd        | None                                 |
| result     |                                      |
| servers    | None                                 |
| status     | not_run                              |
| updated_at | None                                 |
+------------+--------------------------------------+
ciscovim osdmgmt list check-osds
+--------------------------------------+------------+----------+----------
| ID                                   | Action     | Status   | Created
 |
+--------------------------------------+------------+----------+----------
| cd108b85-2678-4aac-b01e-ee05dcd6fd02 | check-osds | Complete | 2019-01-
| adb56a08-fdc5-4810-ac50-4ea6c6b38e3f | check-osds | Complete | 2019-01-|
+--------------------------------------+------------+----------+----------

ciscovim osdmgmt show check-osds --id <id>
+--------------------+-------------+---------------+-----------+---------+
| Message            | Host        | Role          | Server    | State   |
+--------------------+-------------+---------------+-----------+---------+
| Overall OSD Status | i13-25      | block_storage | 15.0.0.11 | Optimal |
|                    | i13-26      | block_storage | 15.0.0.9  | Optimal |
|                    | i13-27-test | block_storage | 15.0.0.4  | Optimal |
|                    |             |               |           |         |
| Number of OSDs     | i13-25      | block_storage | 15.0.0.11 | 10      |
|                    | i13-26      | block_storage | 15.0.0.9  | 10      |
|                    | i13-27-test | block_storage | 15.0.0.4  | 16      |
+--------------------+-------------+---------------+-----------+---------+
+-------------+--------+--------+----+-------------+-----------+-----------
| Host        | OSDs   | Status | ID | HDD Slot    | Path      | Mount
+-------------+--------+--------+----+-------------+-----------+-----------
.
omitted for doc
.
| i13-27-test | osd.2  | up     | 2  |  4 (JBOD)   | /dev/sda1 |
|             | osd.5  | up     | 5  |  3 (JBOD)   | /dev/sdb1 |
|             | osd.8  | up     | 8  |  6 (JBOD)   | /dev/sdc1 |
|             | osd.11 | up     | 11 |  2 (JBOD)   | /dev/sdd1 |
|             | osd.14 | up     | 14 |  5 (JBOD)   | /dev/sde1 |
|             | osd.19 | up     | 19 |  9 (JBOD)   | /dev/sdi1 |
|             | osd.24 | up     | 24 |  10 (JBOD)  | /dev/sdj1 |
|             | osd.27 | up     | 27 |  8 (JBOD)   | /dev/sdl1 |
|             | osd.28 | up     | 28 |  12 (JBOD)  | /dev/sdm1 |
```

```
|              | osd.29 | up     | 29 |  11 (JBOD) | /dev/sdn1 |
|              | osd.30 | up     | 30 |  13 (JBOD) | /dev/sdo1 |
|              | osd.31 | up     | 31 |  17 (JBOD) | /dev/sdp1 |
|              | osd.32 | up     | 32 |  15 (JBOD) | /dev/sdq1 |
|              | osd.33 | up     | 33 |  14 (JBOD) | /dev/sdr1 |
|              | osd.34 | up     | 34 |  16 (JBOD) | /dev/sds1 |
|              | osd.35 | up     | 35 |  7 (JBOD)  | /dev/sdt1 |
+--------------+--------+--------+----+------------+-----------+
```