Red Hat Ceph Storage 3

Operations Guide

Operational tasks for Red Hat Ceph Storage

Last Updated: 2020-06-09

Red Hat Ceph Storage 3 Operations Guide

Operational tasks for Red Hat Ceph Storage

Legal Notice

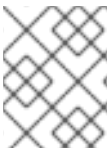
Abstract

This document describes how to do operational tasks for Red Hat Ceph Storage.

Table of Contents

CHAPTER 1. MANAGING THE STORAGE CLUSTER SIZE

As a storage administrator, you can manage the storage cluster size by adding or removing Ceph Monitors or OSDs as storage capacity expands or shrinks.



NOTE

If you are bootstrapping a storage cluster for the first time, see the Red Hat Ceph Storage 3 Installation Guide for Red Hat Enterprise Linux or Ubuntu.

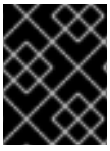
1.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.

1.2. CEPH MONITORS

Ceph monitors are light-weight processes that maintain a master copy of the cluster map. All Ceph clients contact a Ceph monitor and retrieve the current copy of the cluster map, enabling clients to bind to a pool and read and write data.

Ceph monitors use a variation of the Paxos protocol to establish consensus about maps and other critical information across the cluster. Due to the nature of Paxos, Ceph requires a majority of monitors running to establish a quorum thus establishing consensus.



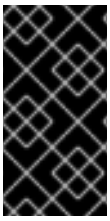
IMPORTANT

Red Hat requires at least three monitors on separate hosts to receive support for a production cluster.

Red Hat recommends deploying an odd number of monitors. An odd number of monitors has a higher resiliency to failures than an even number of monitors. For example, to maintain a quorum on a two monitor deployment, Ceph cannot tolerate any failures; with three monitors, one failure; with four monitors, one failure; with five monitors, two failures. This is why an odd number is advisable. Summarizing, Ceph needs a majority of monitors to be running and to be able to communicate with each other, two out of three, three out of four, and so on.

For an initial deployment of a multi-node Ceph storage cluster, Red Hat requires three monitors, increasing the number two at a time if a valid need for more than three monitors exists.

Since monitors are light-weight, it is possible to run them on the same host as OpenStack nodes. However, Red Hat recommends running monitors on separate hosts.



IMPORTANT

Red Hat does NOT support collocating Ceph Monitors and OSDs on the same node. Doing this can have a negative impact to storage cluster performance.

Red Hat ONLY supports collocating Ceph services in containerized environments.

When you remove monitors from a storage cluster, consider that Ceph monitors use the Paxos protocol to establish a consensus about the master storage cluster map. You must have a sufficient number of monitors to establish a quorum.

Additional Resources

- See the Red Hat Ceph Storage *Supported configurations* Knowledgebase article for all the supported Ceph configurations.

1.2.1. Preparing a new Ceph Monitor node

When adding a new Ceph Monitor to a storage cluster, deploy them on a separate node. The node hardware must be uniform for all monitor nodes in the storage cluster.

Prerequisites

- Network connectivity.
- Having **root** access to the new node.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.

Procedure

1. Add the new node to the server rack.
2. Connect the new node to the network.
3. Install either Red Hat Enterprise Linux 7 or Ubuntu 16.04 on the new node.
4. Install NTP and configure a reliable time source:

```
[root@monitor ~]# yum install ntp
```

5. If using a firewall, open TCP port 6789:

Red Hat Enterprise Linux

```
[root@monitor ~]# firewall-cmd --zone=public --add-port=6789/tcp
[root@monitor ~]# firewall-cmd --zone=public --add-port=6789/tcp --permanent
```

Ubuntu

```
iptables -I INPUT 1 -i $NIC_NAME -p tcp -s $IP_ADDR/$NETMASK_PREFIX --dport 6789 -j ACCEPT
```

Ubuntu example

```
[user@monitor ~]$ sudo iptables -I INPUT 1 -i enp6s0 -p tcp -s 192.168.0.11/24 --dport 6789 -j ACCEPT
```

1.2.2. Adding a Ceph Monitor using Ansible

Red Hat recommends adding two monitors at a time to maintain an odd number of monitors. For example, if you have a three monitors in the storage cluster, Red Hat recommends expanding it to a five monitors.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Having **root** access to the new nodes.

Procedure

1. Add the new Ceph Monitor nodes to the **/etc/ansible/hosts** Ansible inventory file, under a **[mons]** section:

   Example

   ```
   [mons]
   monitor01
   monitor02
   monitor03
   $NEW_MONITOR_NODE_NAME
   $NEW_MONITOR_NODE_NAME
   ```

2. Verify that Ansible can contact the Ceph nodes:

   ```
   # ansible all -m ping
   ```

3. Change directory to the Ansible configuration directory:

   ```
   # cd /usr/share/ceph-ansible
   ```

4. Run the Ansible playbook:

   ```
   $ ansible-playbook site.yml
   ```

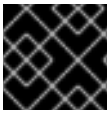
   If adding new monitors to a containerized deployment of Ceph, run the **site-docker.yml** playbook:

   ```
   $ ansible-playbook site-docker.yml
   ```

5. After the Ansible playbook is finish, the new monitor nodes will be in the storage cluster.

1.2.3. Adding a Ceph Monitor using the command-line interface

Red Hat recommends adding two monitors at a time to maintain an odd number of monitors. For example, if you have a three monitors in the storage cluster, Red Hat recommends expanding too five monitors.



IMPORTANT

Red Hat recommends only running one Ceph monitor daemon per node.

Prerequisites

- A running Red Hat Ceph Storage cluster.

- Having **root** access to a running Ceph Monitor node and to the new monitor nodes.

Procedure

1. Add the Red Hat Ceph Storage 3 monitor repository.

   Red Hat Enterprise Linux

   ```
   [root@monitor ~]# subscription-manager repos --enable=rhel-7-server-rhceph-3-mon-rpms
   ```

   Ubuntu

   ```
   [user@monitor ~]$ sudo bash -c 'umask 0077; echo deb
   https://$CUSTOMER_NAME:$CUSTOMER_PASSWORD@rhcs.download.redhat.com/3-
   updates/Tools $(lsb_release -sc) main | tee /etc/apt/sources.list.d/Tools.list'
   [user@monitor ~]$ sudo bash -c 'wget -O - https://www.redhat.com/security/fd431d51.txt |
   apt-key add -'
   ```

2. Install the **ceph-mon** package on the new Ceph Monitor nodes:

   Red Hat Enterprise Linux

   ```
   [root@monitor ~]# yum install ceph-mon
   ```

   Ubuntu

   ```
   [user@monitor ~]$ sudo apt-get install ceph-mon
   ```

3. To ensure the storage cluster identifies the monitor on start or restart, add the monitor's IP address to the Ceph configuration file.
   To add the new monitors to the **[mon]** or **[global]** section of the Ceph configuration file on an existing monitor node in the storage cluster. The **mon_host** setting, which is a list of DNS-resolvable host names or IP addresses, separated by "," or ";" or " ". Optionally, you can also create a specific section in the Ceph configuration file for the new monitor nodes:

   Syntax

   ```
   [mon]
   mon host = $MONITOR_IP:$PORT $MONITOR_IP:$PORT ... $NEW_MONITOR_IP:$PORT
   ```

   or

   ```
   [mon.$MONITOR_ID]
   host = $MONITOR_ID
   mon addr = $MONITOR_IP
   ```

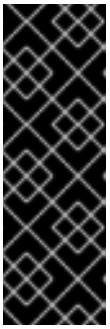
   To make the monitors part of the initial quorum group, you must also add the host name to the **mon_initial_members** parameter in the **[global]** section of the Ceph configuration file.

   Example

   ```
   [global]
   ```

```
mon initial members = node1 node2 node3 node4 node5
...
[mon]
mon host = 192.168.0.1:6789 192.168.0.2:6789 192.168.0.3:6789 192.168.0.4:6789
192.168.0.5:6789
...
[mon.node4]
host = node4
mon addr = 192.168.0.4

[mon.node5]
host = node5
mon addr = 192.168.0.5
```



IMPORTANT

Production storage clusters REQUIRE at least three monitors set in **mon_initial_members** and **mon_host** to ensure high availability. If a storage cluster with only one initial monitor adds two more monitors, but does not add them to **mon_initial_members** and **mon_host**, the failure of the initial monitor will cause the storage cluster to lock up. If the monitors you are adding are replacing monitors that are part of **mon_initial_members** and **mon_host**, the new monitors must be added to **mon_initial_members** and **mon_host** too.

4. Copy the updated Ceph configuration file to all Ceph nodes and Ceph clients:

   Syntax

   ```
   scp /etc/ceph/$CLUSTER_NAME.conf $TARGET_NODE_NAME:/etc/ceph
   ```

   Example

   ```
   [root@monitor ~]# scp /etc/ceph/ceph.conf node4:/etc/ceph
   ```

5. Create the monitor's data directory on the new monitor nodes:

   Syntax

   ```
   mkdir /var/lib/ceph/mon/$CLUSTER_NAME-$MONITOR_ID
   ```

   Example

   ```
   [root@monitor ~]# mkdir /var/lib/ceph/mon/ceph-node4
   ```

6. Create a temporary directory on a running monitor node and on the new monitor nodes to keep the files needed for this procedure. This directory should be different from the monitor's default directory created in the previous step, and can be removed after all the steps are completed:

   Syntax

   ```
   mkdir $TEMP_DIRECTORY
   ```

Exmaple

```
[root@monitor ~]# mkdir /tmp/ceph
```

7. Copy the admin key from a running monitor node to the new monitor nodes so that you can run **ceph** commands:

   Syntax

   ```
   scp /etc/ceph/$CLUSTER_NAME.client.admin.keyring $TARGET_NODE_NAME:/etc/ceph
   ```

   Example

   ```
   [root@monitor ~]# scp /etc/ceph/ceph.client.admin.keyring node4:/etc/ceph
   ```

8. From a running monitor node, retrieve the monitor keyring:

   Syntax

   ```
   ceph auth get mon. -o /$TEMP_DIRECTORY/$KEY_FILE_NAME
   ```

   Example

   ```
   [root@monitor ~]# ceph auth get mon. -o /tmp/ceph/ceph_keyring.out
   ```

9. From a running monitor node, retrieve the monitor map:

   Syntax

   ```
   ceph mon getmap -o /$TEMP_DIRECTORY/$MONITOR_MAP_FILE
   ```

   Example

   ```
   [root@monitor ~]# ceph mon getmap -o /tmp/ceph/ceph_mon_map.out
   ```

10. Copy the collected monitor data to the new monitor nodes:

    Syntax

    ```
    scp /tmp/ceph $TARGET_NODE_NAME:/tmp/ceph
    ```

    Example

    ```
    [root@monitor ~]# scp /tmp/ceph node4:/tmp/ceph
    ```

11. Prepare the new monitors' data directory from the data you collected earlier. You must specify the path to the monitor map to retrieve quorum information from the monitors, along with their **fsid**. You must also specify a path to the monitor keyring:

    Syntax

    -

```
ceph-mon -i $MONITOR_ID --mkfs --monmap
/$TEMP_DIRECTORY/$MONITOR_MAP_FILE --keyring
/$TEMP_DIRECTORY/$KEY_FILE_NAME
```

Example

```
[root@monitor ~]# ceph-mon -i node4 --mkfs --monmap /tmp/ceph/ceph_mon_map.out --
keyring /tmp/ceph/ceph_keyring.out
```

12. For storage clusters with custom names, add the the following line to the **/etc/sysconfig/ceph** file:

Red Hat Enterprise Linux

```
[root@monitor ~]# echo "CLUSTER=<custom_cluster_name>" >> /etc/sysconfig/ceph
```

Ubuntu

```
[user@monitor ~]$ sudo echo "CLUSTER=<custom_cluster_name>" >> /etc/default/ceph
```

13. Update the owner and group permissions on the new monitor nodes:

Syntax

```
chown -R $OWNER:$GROUP $DIRECTORY_PATH
```

Example

```
[root@monitor ~]# chown -R ceph:ceph /var/lib/ceph/mon
[root@monitor ~]# chown -R ceph:ceph /var/log/ceph
[root@monitor ~]# chown -R ceph:ceph /var/run/ceph
[root@monitor ~]# chown -R ceph:ceph /etc/ceph
```

14. Enable and start the **ceph-mon** process on the new monitor nodes:

Syntax

```
systemctl enable ceph-mon.target
systemctl enable ceph-mon@$MONITOR_ID
systemctl start ceph-mon@$MONITOR_ID
```

Example

```
[root@monitor ~]# systemctl enable ceph-mon.target
[root@monitor ~]# systemctl enable ceph-mon@node4
[root@monitor ~]# systemctl start ceph-mon@node4
```

Additional Resources

- See the *Enabling the Red Hat Ceph Storage Repositories* section in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.

1.2.4. Removing a Ceph Monitor using Ansible

To remove a Ceph Monitor with Ansible, use the **shrink-mon.yml** playbook.

Prerequisites

- An Ansible administration node.
- A running Red Hat Ceph Storage cluster deployed by Ansible.

Procedure

1. Change to the **/usr/share/ceph-ansible/** directory.

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

2. Copy the **shrink-mon.yml** playbook from the **infrastructure-playbooks** directory to the current directory.

```
[root@admin ceph-ansible]# cp infrastructure-playbooks/shrink-mon.yml .
```

3. Run the **shrink-mon.yml** playbook for either normal or containerized deployments of Red Hat Ceph Storage:

```
[user@admin ceph-ansible]$ ansible-playbook shrink-mon.yml -e mon_to_kill=<hostname> -u <ansible-user>
```

Replace:

- **<hostname>** with the short host name of the Monitor node. To remove more Monitors, separate their host names with comma.
- **<ansible-user>** with the name of the Ansible user

For example, to remove a Monitor that is located on a node with **monitor1** host name:

```
[user@admin ceph-ansible]$ ansible-playbook shrink-mon.yml -e mon_to_kill=monitor1 -u user
```

4. Remove the Monitor entry from all Ceph configuration files in the cluster.
5. Ensure that the Monitor has been successfully removed.

```
[root@monitor ~]# ceph -s
```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Installation Guide for Red Hat Enterprise Linux* or Ubuntu.

1.2.5. Removing a Ceph Monitor using the command-line interface

Removing a Ceph Monitor involves removing a **ceph-mon** daemon from the storage cluster and updating the storage cluster map.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Having **root** access to the monitor node.

Procedure

1. Stop the monitor service:

   Syntax

   ```
   systemctl stop ceph-mon@$MONITOR_ID
   ```

   Example

   ```
   [root@monitor ~]# systemctl stop ceph-mon@node3
   ```

2. Remove the monitor from the storage cluster:

   Syntax

   ```
   ceph mon remove $MONITOR_ID
   ```

   Example

   ```
   [root@monitor ~]# ceph mon remove node3
   ```

3. Remove the monitor entry from the Ceph configuration file, by default **/etc/ceph/ceph.conf**.

4. Redistribute the Ceph configuration file to all remaining Ceph nodes in the storage cluster:

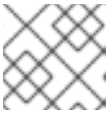
   Syntax

   ```
   scp /etc/ceph/$CLUSTER_NAME.conf $USER_NAME@$TARGET_NODE_NAME:/etc/ceph/
   ```

   Example

   ```
   [root@monitor ~]# scp /etc/ceph/ceph.conf root@$node1:/etc/ceph/
   ```

5. Containers only. Disable the monitor service:

   NOTE

   Perform steps 5-9 only if using containers.

   Syntax

```
systemctl disable ceph-mon@$MONITOR_ID
```

Example

```
[root@monitor ~]# systemctl disable ceph-mon@node3
```

6. Containers only. Remove the service from systemd:

```
[root@monitor ~]# rm /etc/systemd/system/ceph-mon@.service
```

7. Containers only. Reload the systemd manager configuration:

```
[root@monitor ~]# systemctl daemon-reload
```

8. Containers only. Reset the state of the failed monitor unit:

```
[root@monitor ~]# systemctl reset-failed
```

9. Containers only. Remove the **ceph-mon** RPM:

```
[root@monitor ~]# docker exec node3 yum remove ceph-mon
```

10. Optionally, you can archive the monitor data:

Syntax

```
mv /var/lib/ceph/mon/$CLUSTER_NAME-$MONITOR_ID /var/lib/ceph/mon/removed-$CLUSTER_NAME-$MONITOR_ID
```

Example

```
[root@monitor ~]# mv /var/lib/ceph/mon/ceph-node3 /var/lib/ceph/mon/removed-ceph-node3
```

11. Optionally, you can delete the monitor data:

Syntax

```
rm -r /var/lib/ceph/mon/$CLUSTER_NAME-$MONITOR_ID
```

Example

```
[root@monitor ~]# rm -r /var/lib/ceph/mon/ceph-node3
```

1.2.6. Removing a Ceph Monitor from an unhealthy storage cluster

This procedure removes a **ceph-mon** daemon from an unhealthy storage cluster. An unhealthy storage cluster that has placement groups persistently not **active + clean**.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Having **root** access to the monitor node.

Procedure

1. Identify a surviving Ceph Monitor and log in to that node:

```
[root@monitor ~]# ceph mon dump
[root@monitor ~]# ssh $MONITOR_HOST_NAME
```

WARNING

For the **ceph mod dump** command to return data, there must be quorum with the Ceph Monitors.

2. Stop the **ceph-mon** daemon and extract a copy of the **monmap** file. :

Syntax

```
systemctl stop ceph-mon@$MONITOR_ID
ceph-mon -i $MONITOR_ID --extract-monmap $TEMPORARY_PATH
```

Example

```
[root@monitor ~]# systemctl stop ceph-mon@node1
[root@monitor ~]# ceph-mon -i node1 --extract-monmap /tmp/monmap
```

3. Remove the non-surviving monitor(s):

Syntax

```
monmaptool $TEMPORARY_PATH --rm $MONITOR_ID
```

Example

```
[root@monitor ~]# monmaptool /tmp/monmap --rm node2
```

4. Inject the surviving monitor map with the removed monitor(s) into the surviving monitor:

Syntax

```
ceph-mon -i $MONITOR_ID --inject-monmap $TEMPORARY_PATH
```

Example

```
[root@monitor ~]# ceph-mon -i node1 --inject-monmap /tmp/monmap
```

1.3. CEPH OSDS

When a Red Hat Ceph Storage cluster is up and running, you can add OSDs to the storage cluster at runtime.

A Ceph OSD generally consists of one **ceph-osd** daemon for one storage drive and its associated journal within a node. If a node has multiple storage drives, then map one **ceph-osd** daemon for each drive.

Red Hat recommends checking the capacity of a cluster regularly to see if it is reaching the upper end of its storage capacity. As a storage cluster reaches its **near full** ratio, add one or more OSDs to expand the storage cluster's capacity.

When you want to reduce the size of a Red Hat Ceph Storage cluster or replace the hardware, you can also remove an OSD at runtime. If the node has multiple storage drives, you might also need to remove one of the **ceph-osd** daemon for that drive. Generally, it's a good idea to check the capacity of the storage cluster to see if you are reaching the upper end of its capacity. Ensure that when you remove an OSD that the storage cluster is not at its **near full** ratio.

IMPORTANT

Do not let a storage cluster reach the **full** ratio before adding an OSD. OSD failures that occur after the storage cluster reaches the **near full** ratio can cause the storage cluster to exceed the **full** ratio. Ceph blocks write access to protect the data until you resolve the storage capacity issues. Do not remove OSDs without considering the impact on the **full** ratio first.

1.3.1. Ceph OSD node configuration

Ceph OSDs and their supporting hardware should be similarly configured as a storage strategy for the pool(s) that will use the OSDs. Ceph prefers uniform hardware across pools for a consistent performance profile. For best performance, consider a CRUSH hierarchy with drives of the same type or size. See the Storage Strategies guide for more details.

If you add drives of dissimilar size, then you will need to adjust their weights accordingly. When you add the OSD to the CRUSH map, consider the weight for the new OSD. Hard drive capacity grows approximately 40% per year, so newer OSD nodes might have larger hard drives than older nodes in the storage cluster, that is, they might have a greater weight.

Before doing a new installation, review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.

1.3.2. Mapping a container OSD ID to a drive

Sometimes it is necessary to identify which drive a containerized OSD is using. For example, if an OSD has an issue you might need to know which drive it uses to verify the drive status. Also, for a non-containerized OSD you reference the OSD ID to start and stop it, but to start and stop a containerized OSD you must reference the drive it uses.

Prerequisites

- A running Red Hat Ceph Storage cluster in a containerized environment.
- Having **root** access to the container host.

Procedure

1. Find a container name. For example, to identify the drive associated with **osd.5**, open a terminal on the container node where **osd.5** is running, and then run **docker ps** to list all containers:

   Example

```
[root@ceph3 ~]# docker ps
CONTAINER ID        IMAGE                                                 COMMAND             CREATED
STATUS              PORTS               NAMES
3a866f927b74        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
About an hour ago   Up About an hour                 ceph-osd-ceph3-sdd
91f3d4829079        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
22 hours ago        Up 22 hours                      ceph-osd-ceph3-sdb
73dfe4021a49        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
7 days ago          Up 7 days                        ceph-osd-ceph3-sdf
90f6d756af39        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
7 days ago          Up 7 days                        ceph-osd-ceph3-sde
e66d6e33b306        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
7 days ago          Up 7 days                        ceph-mgr-ceph3
733f37aafd23        registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest   "/entrypoint.sh"
7 days ago          Up 7 days                        ceph-mon-ceph3
```

2. Use **docker exec** to run **ceph-volume lvm list** on any OSD container name from the previous output:

   Example

```
[root@ceph3 ~]# docker exec ceph-osd-ceph3-sdb ceph-volume lvm list
====== osd.5 =======

  [journal]    /dev/journals/journal1

      journal uuid              C65n7d-B1gy-cqX3-vZKY-ZoE0-IEYM-HnIJzs
      osd id                    1
      cluster fsid              ce454d91-d748-4751-a318-ff7f7aa18ffd
      type                      journal
      osd fsid                  661b24f8-e062-482b-8110-826ffe7f13fa
      data uuid                 SIEgHe-jX1H-QBQk-Sce0-RUls-8KlY-g8HgcZ
      journal device            /dev/journals/journal1
      data device               /dev/test_group/data-lv2
      devices                   /dev/sda

  [data]    /dev/test_group/data-lv2

      journal uuid              C65n7d-B1gy-cqX3-vZKY-ZoE0-IEYM-HnIJzs
      osd id                    1
      cluster fsid              ce454d91-d748-4751-a318-ff7f7aa18ffd
      type                      data
      osd fsid                  661b24f8-e062-482b-8110-826ffe7f13fa
      data uuid                 SIEgHe-jX1H-QBQk-Sce0-RUls-8KlY-g8HgcZ
      journal device            /dev/journals/journal1
      data device               /dev/test_group/data-lv2
      devices                   /dev/sdb
```

From this output you can see **osd.5** is associated with **/dev/sdb**.

Additional Resources

- See Replacing a failed OSD disk for more information

1.3.3. Adding a Ceph OSD using Ansible with the same disk topology

For Ceph OSDs with the same disk topology, Ansible will add the same number of OSDs as other OSD nodes using the same device paths specified in the **devices:** section of the **/usr/share/ceph-ansible/group_vars/osds** file.

NOTE

The new Ceph OSD node(s) will have the same configuration as the rest of the OSDs.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.
- Having **root** access to the new nodes.
- The same number of OSD data drives as other OSD nodes in the storage cluster.

Procedure

1. Add the Ceph OSD node(s) to the **/etc/ansible/hosts** file, under the **[osds]** section:

   Example

   ```
   [osds]
   ...
   osd06
   $NEW_OSD_NODE_NAME
   ```

2. Verify that Ansible can reach the Ceph nodes:

   ```
   [user@admin ~]$ ansible all -m ping
   ```

3. Navigate to the Ansible configuration directory:

   ```
   [user@admin ~]$ cd /usr/share/ceph-ansible
   ```

4. Copy the **add-osd.yml** file to the **/usr/share/ceph-ansible/** directory:

   ```
   [user@admin ceph-ansible]$ sudo cp infrastructure-playbooks/add-osd.yml .
   ```

5. Run the Ansible playbook for either normal or containerized deployments of Ceph:

   ```
   [user@admin ceph-ansible]$ ansible-playbook add-osd.yml
   ```

NOTE

When adding an OSD, if the playbook fails with **PGs were not reported as active+clean**, configure the following variables in the **all.yml** file to adjust the retries and delay:

```
# OSD handler checks
handler_health_osd_check_retries: 50
handler_health_osd_check_delay: 30
```

1.3.4. Adding a Ceph OSD using Ansible with different disk topologies

For Ceph OSDs with different disk topologies, there are two approaches for adding the new OSD node(s) to an existing storage cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.
- Having **root** access to the new nodes.

Procedure

1. **First Approach**

   a. Add the new Ceph OSD node(s) to the **/etc/ansible/hosts** file, under the **[osds]** section:

   Example

   ```
   [osds]
   ...
   osd06
   $NEW_OSD_NODE_NAME
   ```

   b. Create a new file for each new Ceph OSD node added to the storage cluster, under the **/etc/ansible/host_vars/** directory:

   Syntax

   ```
   touch /etc/ansible/host_vars/$NEW_OSD_NODE_NAME
   ```

   Example

   ```
   [root@admin ~]# touch /etc/ansible/host_vars/osd07
   ```

   c. Edit the new file, and add the **devices:** and **dedicated_devices:** sections to the file. Under each of these sections add a **-**, space, then the full path to the block device names for this OSD node:

   Example

```
devices:
  - /dev/sdc
  - /dev/sdd
  - /dev/sde
  - /dev/sdf

dedicated_devices:
  - /dev/sda
  - /dev/sda
  - /dev/sdb
  - /dev/sdb
```

d. Verify that Ansible can reach all the Ceph nodes:

```
[user@admin ~]$ ansible all -m ping
```

e. Change directory to the Ansible configuration directory:

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

f. Copy the **add-osd.yml** file to the **/usr/share/ceph-ansible/** directory:

```
[user@admin ceph-ansible]$ sudo cp infrastructure-playbooks/add-osd.yml .
```

g. Run the Ansible playbook:

```
[user@admin ceph-ansible]$ ansible-playbook add-osd.yml
```

2. **Second Approach**

a. Add the new OSD node name to the **/etc/ansible/hosts** file, and use the **devices** and **dedicated_devices** options, specifying the different disk topology:

Example

```
[osds]
...
osd07 devices="['/dev/sdc', '/dev/sdd', '/dev/sde', '/dev/sdf']" dedicated_devices="
['/dev/sda', '/dev/sda', '/dev/sdb', '/dev/sdb']"
```

b. Verify that Ansible can reach the all Ceph nodes:

```
[user@admin ~]$ ansible all -m ping
```

c. Change directory to the Ansible configuration directory:

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

d. Copy the **add-osd.yml** file to the **/usr/share/ceph-ansible/** directory:

```
[user@admin ceph-ansible]$ sudo cp infrastructure-playbooks/add-osd.yml .
```

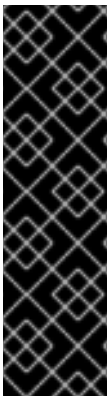
e. Run the Ansible playbook:

```
[user@admin ceph-ansible]$ ansible-playbook add-osd.yml
```

1.3.5. Adding a Ceph OSD using the command-line interface

Here is the high-level workflow for manually adding an OSD to a Red Hat Ceph Storage:

1. Install the **ceph-osd** package and create a new OSD instance
2. Prepare and mount the OSD data and journal drives
3. Add the new OSD node to the CRUSH map
4. Update the owner and group permissions
5. Enable and start the **ceph-osd** daemon



IMPORTANT

The **ceph-disk** command is deprecated. The **ceph-volume** command is now the preferred method for deploying OSDs from the command-line interface. Currently, the **ceph-volume** command only supports the **lvm** plugin. Red Hat will provide examples throughout this guide using both commands as a reference, allowing time for storage administrators to convert any custom scripts that rely on **ceph-disk** to **ceph-volume** instead.

See the Red Hat Ceph Storage Administration Guide, for more information on using the **ceph-volume** command.

NOTE

For custom storage cluster names, use the **--cluster $CLUSTER_NAME** option with the **ceph** and **ceph-osd** commands.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.
- Having **root** access to the new nodes.

Procedure

1. Enable the Red Hat Ceph Storage 3 OSD software repository.

   Red Hat Enterprise Linux

   ```
   [root@osd ~]# subscription-manager repos --enable=rhel-7-server-rhceph-3-osd-rpms
   ```

   Ubuntu

```
[user@osd ~]$ sudo bash -c 'umask 0077; echo deb
https://customername:customerpasswd@rhcs.download.redhat.com/3-updates/Tools
$(lsb_release -sc) main | tee /etc/apt/sources.list.d/Tools.list'
[user@osd ~]$ sudo bash -c 'wget -O - https://www.redhat.com/security/fd431d51.txt | apt-
key add -'
```

2. Create the **/etc/ceph/** directory:

```
# mkdir /etc/ceph
```

3. On the new OSD node, copy the Ceph administration keyring and configuration files from one of the Ceph Monitor nodes:

   Syntax

```
scp
$USER_NAME@$MONITOR_HOST_NAME:/etc/ceph/$CLUSTER_NAME.client.admin.keyrin
g /etc/ceph
scp $USER_NAME@$MONITOR_HOST_NAME:/etc/ceph/$CLUSTER_NAME.conf
/etc/ceph
```

   Example

```
[root@osd ~]# scp root@node1:/etc/ceph/ceph.client.admin.keyring /etc/ceph/
[root@osd ~]# scp root@node1:/etc/ceph/ceph.conf /etc/ceph/
```

4. Install the **ceph-osd** package on the new Ceph OSD node:

   Red Hat Enterprise Linux

```
[root@osd ~]# yum install ceph-osd
```

   Ubuntu

```
[user@osd ~]$ sudo apt-get install ceph-osd
```

5. Decide if you want to collocate a journal or use a dedicated journal for the new OSDs.

NOTE

The **--filestore** option is required.

   a. For OSDs with a collocated journal:

      Syntax

```
[root@osd ~]# ceph-disk --setuser ceph --setgroup ceph prepare  --filestore
/dev/$DEVICE_NAME
```

      Examples

```
[root@osd ~]# ceph-disk --setuser ceph --setgroup ceph prepare  --filestore /dev/sda
```

b. For OSDs with a dedicated journal:

Syntax

```
[root@osd ~]# ceph-disk --setuser ceph --setgroup ceph prepare  --filestore
/dev/$DEVICE_NAME /dev/$JOURNAL_DEVICE_NAME
```

or

```
[root@osd ~]# ceph-volume lvm prepare  --filestore --data /dev/$DEVICE_NAME --
journal /dev/$JOURNAL_DEVICE_NAME
```

Examples

```
[root@osd ~]# ceph-disk --setuser ceph --setgroup ceph prepare  --filestore /dev/sda
/dev/sdb
```

```
[root@osd ~]# ceph-volume lvm prepare  --filestore --data /dev/vg00/lvol1 --journal
/dev/sdb
```

6. Set the **noup** option:

```
[root@osd ~]# ceph osd set noup
```

7. Activate the new OSD:

Syntax

```
[root@osd ~]# ceph-disk activate /dev/$DEVICE_NAME
```

or

```
[root@osd ~]# ceph-volume lvm activate --filestore $OSD_ID $OSD_FSID
```

Example

```
[root@osd ~]# ceph-disk activate /dev/sda
```

```
[root@osd ~]# ceph-volume lvm activate --filestore 0 6cc43680-4f6e-4feb-92ff-
9c7ba204120e
```

8. Add the OSD to the CRUSH map:

Syntax

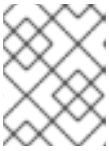
```
ceph osd crush add $OSD_ID $WEIGHT [$BUCKET_TYPE=$BUCKET_NAME ...]
```

Example

```
[root@osd ~]# ceph osd crush add 4 1 host=node4
```

NOTE

If you specify more than one bucket, the command places the OSD into the most specific bucket out of those you specified, *and* it moves the bucket underneath any other buckets you specified.



NOTE

You can also edit the CRUSH map manually. See the Editing a CRUSH map section in the Storage Strategies guide for Red Hat Ceph Storage 3.

IMPORTANT

If you specify only the root bucket, then the OSD attaches directly to the root, but the CRUSH rules expect OSDs to be inside of the host bucket.

9. Unset the **noup** option:

```
[root@osd ~]# ceph osd unset noup
```

10. Update the owner and group permissions for the newly created directories:

Syntax

```
chown -R $OWNER:$GROUP $PATH_TO_DIRECTORY
```

Example

```
[root@osd ~]# chown -R ceph:ceph /var/lib/ceph/osd
[root@osd ~]# chown -R ceph:ceph /var/log/ceph
[root@osd ~]# chown -R ceph:ceph /var/run/ceph
[root@osd ~]# chown -R ceph:ceph /etc/ceph
```

11. If you use clusters with custom names, then add the following line to the appropriate file:

Red Hat Enterprise Linux

```
[root@osd ~]# echo "CLUSTER=$CLUSTER_NAME" >> /etc/sysconfig/ceph
```

Ubuntu

```
[user@osd ~]$ sudo echo "CLUSTER=$CLUSTER_NAME" >> /etc/default/ceph
```

Replace **$CLUSTER_NAME** with the custom cluster name.

12. To ensure that the new OSD is **up** and ready to receive data, enable and start the OSD service:

Syntax

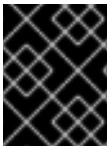
```
systemctl enable ceph-osd@$OSD_ID
systemctl start ceph-osd@$OSD_ID
```

Example

```
[root@osd ~]# systemctl enable ceph-osd@4
[root@osd ~]# systemctl start ceph-osd@4
```

1.3.6. Removing a Ceph OSD using Ansible

At times, you might need to scale down the capacity of a Red Hat Ceph Storage cluster. To remove an OSD from a Red Hat Ceph Storage cluster using Ansible, and depending on which OSD scenario is used, run the **shrink-osd.yml** or **shrink-osd-ceph-disk.yml** playbook. If **osd_scenario** is set to **collocated** or **non-collocated**, then use the **shrink-osd-ceph-disk.yml** playbook. If **osd_scenario** is set to **lvm**, then use the **shrink-osd.yml** playbook.



IMPORTANT

Removing an OSD from the storage cluster will destroy all the data contained on that OSD.

Prerequisites

- A running Red Hat Ceph Storage deployed by Ansible.
- A running Ansible administration node.
- Root-level access to the Ansible administration node.

Procedure

1. Change to the **/usr/share/ceph-ansible/** directory.

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

2. Copy the appropriate playbook from the **infrastructure-playbooks** directory to the current directory.

```
[root@admin ceph-ansible]# cp infrastructure-playbooks/shrink-osd.yml .
```

or

```
[root@admin ceph-ansible]# cp infrastructure-playbooks/shrink-osd-ceph-disk.yml .
```

3. For **bare-metal** or **containers** deployments, run the appropriate Ansible playbook:

Syntax

```
ansible-playbook shrink-osd.yml -e osd_to_kill=$ID -u $ANSIBLE_USER
```

or

```
ansible-playbook shrink-osd-ceph-disk.yml -e osd_to_kill=$ID -u $ANSIBLE_USER
```

Replace:

- **$ID** with the ID of the OSD. To remove more OSDs, separate the OSD IDs with a comma.
- **$ANSIBLE_USER** with the name of the Ansible user

Example

```
[user@admin ceph-ansible]$ ansible-playbook shrink-osd.yml -e osd_to_kill=1 -u user
```

or

```
[user@admin ceph-ansible]$ ansible-playbook shrink-osd-ceph-disk.yml -e osd_to_kill=1 -u user
```

4. Verify that the OSD has been successfully removed:

```
[root@mon ~]# ceph osd tree
```

Additional Resources

- See the *Red Hat Ceph Storage Installation Guide* for *Red Hat Enterprise Linux* or *Ubuntu* for details.

1.3.7. Removing a Ceph OSD using the command-line interface

Removing an OSD from a storage cluster involves updating the cluster map, removing its authentication key, removing the OSD from the OSD map, and removing the OSD from the **ceph.conf** file. If the node has multiple drives, you might need to remove an OSD for each drive by repeating this procedure.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Enough available OSDs so that the storage cluster is not at its **near full** ratio.
- Having **root** access to the OSD node.

Procedure

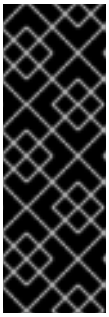
1. Remove the OSD from the storage cluster:

   Syntax

   ```
   ceph osd out $OSD_ID
   ```

   Example

   ```
   [root@osd ~]# ceph osd out 4
   ```



IMPORTANT

Once the OSD is out, Ceph will start rebalancing and copying data to other OSDs in the storage cluster. Red Hat recommends waiting until the storage cluster becomes **active+clean** before proceeding to the next step. To observe the data migration, run the following command:

```
[root@monitor ~]# ceph -w
```

2. Disable and stop the OSD service:

Syntax

```
systemctl disable ceph-osd@$OSD_ID
systemctl stop ceph-osd@$OSD_ID
```

Example

```
[root@osd ~]# systemctl disable ceph-osd@4
[root@osd ~]# systemctl stop ceph-osd@4
```

Once the OSD is stopped, it is **down**.

3. Remove the OSD from the CRUSH map so that it no longer receives data.

Syntax

```
ceph osd crush remove $OSD_NAME
```

Example

```
[root@osd ~]# ceph osd crush remove osd.4
```

NOTE

You can also decompile the CRUSH map, remove the OSD from the device list, remove the device as an item in the host bucket or remove the host bucket. If it is in the CRUSH map and you intend to remove the host, recompile the map and set it. See the Storage Strategies Guide for details.

4. Remove the OSD authentication key:

Syntax

```
ceph auth del osd.$OSD_ID
```

Example

```
[root@osd ~]# ceph auth del osd.4
```

5. Remove the OSD:

Syntax

```
ceph osd rm $OSD_ID
```

Example

```
[root@osd ~]# ceph osd rm 4
```

6. Edit the storage cluster's configuration file, by default **/etc/ceph.conf**, and remove the OSD entry, if it exists:

Example

```
[osd.4]
host = $HOST_NAME
```

7. Remove the reference to the OSD in the **/etc/fstab** file, if the OSD was added manually.

8. Copy the updated configuration file to the **/etc/ceph/** directory of all other nodes in the storage cluster.

Syntax

```
scp /etc/ceph/$CLUSTER_NAME.conf $USER_NAME@$HOST_NAME:/etc/ceph/
```

Example

```
[root@osd ~]# scp /etc/ceph/ceph.conf root@node4:/etc/ceph/
```

1.3.8. Replacing a journal using the command-line interface

The procedure to replace a journal when the journal and data devices are on the same physical device, for example when using **osd_scenario: collocated**, requires replacing the whole OSD. However, on an OSD where the journal is on a separate physical device from the data device, for example when using **osd_scenario: non-collocated**, you can replace just the journal device.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A new partition or storage device.

Procedure

1. Set the cluster to **noout** to prevent backfilling:

```
[root@osd1 ~]# ceph osd set noout
```

2. Stop the OSD where the journal will be changed:

```
[root@osd1 ~]# systemctl stop ceph-osd@$OSD_ID
```

3. Flush the journal on the OSD:

```
[root@osd1 ~]# ceph-osd -i $OSD_ID --flush-journal
```

4. Remove the old journal partition to prevent partition UUID collision with the new partition:

```
sgdisk --delete=$OLD_PART_NUM -- $OLD_DEV_PATH
```

Replace

- **$OLD_PART_NUM** with the partition number of the old journal device.
- **$OLD_DEV_PATH** with the path to the old journal device.

Example

```
[root@osd1 ~]# sgdisk --delete=1 -- /dev/sda
```

5. Create the new journal partition on the new device. This **sgdisk** command will use the next available partition number automatically:

```
sgdisk --new=0:0:$JOURNAL_SIZE -- $NEW_DEV_PATH
```

Replace

- **$JOURNAL_SIZE** with the journal size appropriate for the environment, for example **10240M**.
- **NEW_DEV_PATH** with the path to the device to be used for the new journal.

NOTE

The minimum default size for a journal is 5 GB. Values over 10 GB are typically not needed. For additional details, contact Red Hat Support.

Example

```
[root@osd1 ~]# sgdisk --new=0:0:10240M -- /dev/sda
```

6. Set the proper parameters on the new partition:

```
sgdisk --change-name=0:"ceph journal" --partition-guid=0:$OLD_PART_UUID --
typecode=0:45b0969e-9b03-4f30-b4c6-b4b80ceff106 --mbrtogpt -- $NEW_DEV_PATH
```

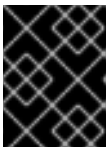
Replace

- **$OLD_PART_UUID** with UUID in the **journal_uuid** file for the relevant OSD. For example, for OSD **0** use the UUID in **/var/lib/ceph/osd/ceph-0/journal_uuid**.
- **NEW_DEV_PATH** with the path to the device to be used for the new journal.

Example

```
[root@osd1 ~]# sgdisk --change-name=0:"ceph journal" --partition-guid=0:a1279726-a32d-4101-880d-e8573bb11c16 --typecode=0:097c058d-0758-4199-a787-ce9bacb13f48 -- mbrtogpt -- /dev/sda
```

After running the above **sgdisk** commands the new journal partition is prepared for Ceph and the journal can be created on it.



IMPORTANT

This command cannot be combined with the partition creation command due to limitations in **sgdisk** causing the partition to not be created correctly.

7. Create the new journal:

```
[root@osd1 ~]# ceph-osd -i $OSD_ID --mkjournal
```

8. Start the OSD:

```
[root@osd1 ~]# systemctl start ceph-osd@$OSD_ID
```

1. Remove the **noout** flag on the OSDs:

```
[root@osd1 ~]# ceph osd unset noout
```

2. Confirm the journal is associated with the correct device:

```
[root@osd1 ~]# ceph-disk list
```

1.3.9. Observing the data migration

When you add or remove an OSD to the CRUSH map, Ceph begins rebalancing the data by migrating placement groups to the new or existing OSD(s).

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Recently added or removed an OSD.

Procedure

1. To observe the data migration:

```
[root@monitor ~]# ceph -w
```

2. Watch as the placement group states change from **active+clean** to **active, some degraded objects**, and finally **active+clean** when migration completes.

3. To exit the utility, press **Ctrl + C**.

1.4. RECALCULATING THE PLACEMENT GROUPS

Placement groups (PGs) define the spread of any pool data across the available OSDs. A placement group is build upon the given redundancy algorithm to be used. For a 3-way replication, the redundancy is defined to use three different OSDs. For erasure-coded pools, the number of OSDs to use is defined by the number of chunks.

When defining a pool the number of placement groups defines the grade of granularity the data is spread with across all available OSDs. The higher the number the better the equalization of capacity load can be. However, since handling the placement groups is also important in case of reconstruction of data, the number is significant to be carefully chosen upfront. To support calculation a tool is available to produce agile environments.

During lifetime of a storage cluster a pool may grow above the initially anticipated limits. With the growing number of drives a recalculation is recommended. The number of placement groups per OSD should be around 100. When adding more OSDs to the storage cluster the number of PGs per OSD will lower over time. Starting with 120 drives initially in the storage cluster and setting the **pg_num** of the pool to 4000 will end up in 100 PGs per OSD, given with the replication factor of three. Over time, when growing to ten times the number of OSDs, the number of PGs per OSD will go down to ten only. Because small number of PGs per OSD will tend to an unevenly distributed capacity, consider adjusting the PGs per pool.

Adjusting the number of placement groups can be done online. Recalculating is not only a recalculation of the PG numbers, but will involve data relocation, which will be a lengthy process. However, the data availability will be maintained at any time.

Very high numbers of PGs per OSD should be avoided, because reconstruction of all PGs on a failed OSD will start at once. A high number of IOPS is required to perform reconstruction in a timely manner, which might not be available. This would lead to deep I/O queues and high latency rendering the storage cluster unusable or will result in long healing times.

Additional Resources

- See the PG calculator for calculating the values by a given use case.
- See the Erasure Code Pools chapter in the Red Hat Ceph Storage Strategies Guide for more information.

1.5. USING THE CEPH MANAGER BALANCER MODULE

The balancer is a module for Ceph Manager that optimizes the placement of placement groups, or PGs, across OSDs in order to achieve a balanced distribution, either automatically or in a supervised fashion.

Prerequisites

- A running Red Hat Ceph Storage cluster

Start the balancer

1. Ensure the balancer module is enabled:

```
[root@mon ~]# ceph mgr module enable balancer
```

2. Turn on the balancer module:

```
[root@mon ~]# ceph balancer on
```

Modes

There are currently two supported balancer modes:

- **crush-compat**: The CRUSH compat mode uses the compat **weight-set** feature, introduced in Ceph Luminous, to manage an alternative set of weights for devices in the CRUSH hierarchy. The normal weights should remain set to the size of the device to reflect the target amount of data that you want to store on the device. The balancer then optimizes the **weight-set** values, adjusting them up or down in small increments in order to achieve a distribution that matches the target distribution as closely as possible. Because PG placement is a pseudorandom process, there is a natural amount of variation in the placement; by optimizing the weights, the balancer counter-acts that natural variation.
  This mode is fully backwards compatible with older clients. When an OSDMap and CRUSH map are shared with older clients, the balancer presents the optimized weights as the real weights.

  The primary restriction of this mode is that the balancer cannot handle multiple CRUSH hierarchies with different placement rules if the subtrees of the hierarchy share any OSDs. Because this configuration makes managing space utilization on the shared OSDs difficult, it is generally not recommended. As such, this restriction is normally not an issue.

- **upmap**: Starting with Luminous, the OSDMap can store explicit mappings for individual OSDs as exceptions to the normal CRUSH placement calculation. These **upmap** entries provide fine-grained control over the PG mapping. This CRUSH mode will optimize the placement of individual PGs in order to achieve a balanced distribution. In most cases, this distribution is "perfect," with an equal number of PGs on each OSD +/-1 PG, as they might not divide evenly.

IMPORTANT

Using upmap requires that all clients be running Red Hat Ceph Storage 3.x or later, and Red Hat Enterprise Linux 7.5 or later.

To allow use of this feature, you must tell the cluster that it only needs to support luminous or later clients with:

```
[root@admin ~]# ceph osd set-require-min-compat-client luminous
```

This command fails if any pre-luminous clients or daemons are connected to the monitors.

Due to a known issue, kernel CephFS clients report themselves as jewel clients. To work around this issue, use the **--yes-i-really-mean-it** flag:

```
[root@admin ~]# ceph osd set-require-min-compat-client luminous --yes-i-really-mean-it
```

You can check what client versions are in use with:

```
[root@admin ~]# ceph features
```

WARNING

In Red Hat Ceph Storage 3.x, the upmap feature is only supported for use by the Ceph Manager balancer module for balancing of PGs as the cluster is used. Manual rebalancing of PGs using the upmap feature is not supported in Red Hat Ceph Storage 3.x.

The default mode is **crush-compat**. The mode can be changed with:

```
[root@mon ~]# ceph balancer mode upmap
```

or:

```
[root@mon ~]# ceph balancer mode crush-compat
```

Status

The current status of the balancer can be checked at any time with:

```
[root@mon ~]# ceph balancer status
```

Automatic balancing

By default, when turning on the balancer module, automatic balancing is used:

```
[root@mon ~]# ceph balancer on
```

The balancer can be turned back off again with:

```
[root@mon ~]# ceph balancer off
```

This will use the **crush-compat** mode, which is backward compatible with older clients and will make small changes to the data distribution over time to ensure that OSDs are equally utilized.

Throttling

No adjustments will be made to the PG distribution if the cluster is degraded, for example, if an OSD has failed and the system has not yet healed itself.

When the cluster is healthy, the balancer throttles its changes such that the percentage of PGs that are misplaced, or need to be moved, is below a threshold of 5% by default. This percentage can be adjusted using the **max_misplaced** setting. For example, to increase the threshold to 7%:

```
[root@mon ~]# ceph config-key set mgr/balancer/max_misplaced .07
```

Supervised optimization

The balancer operation is broken into a few distinct phases:

1. Building a **plan**

2. Evaluating the quality of the data distribution, either for the current PG distribution, or the PG distribution that would result after executing a **plan**

3. Executing the **plan**

   - To evaluate and score the current distribution:

     ```
     [root@mon ~]# ceph balancer eval
     ```

   - To evaluate the distribution for a single pool:

     ```
     [root@mon ~]# ceph balancer eval <pool-name>
     ```

   - To see greater detail for the evaluation:

     ```
     [root@mon ~]# ceph balancer eval-verbose ...
     ```

   - To generate a plan using the currently configured mode:

     ```
     [root@mon ~]# ceph balancer optimize <plan-name>
     ```

     Replace **<plan-name>** with a custom plan name.

   - To see the contents of a plan:

     ```
     [root@mon ~]# ceph balancer show <plan-name>
     ```

   - To discard old plans:

     ```
     [root@mon ~]# ceph balancer rm <plan-name>
     ```

   - To see currently recorded plans use the status command:

     ```
     [root@mon ~]# ceph balancer status
     ```

   - To calculate the quality of the distribution that would result after executing a plan:

     ```
     [root@mon ~]# ceph balancer eval <plan-name>
     ```

   - To execute the plan:

     ```
     [root@mon ~]# ceph balancer execute <plan-name>
     ```

     [NOTE]: Only execute the plan if is expected to improve the distribution. After execution, the plan will be discarded.

1.6. ADDITIONAL RESOURCES

- See the Placement Groups (PGs) chapter in the Red Hat Ceph Storage Strategies Guide for more information.

CHAPTER 2. HANDLING A DISK FAILURE

As a storage administrator, you will have to deal with a disk failure at some point over the life time of the storage cluster. Testing and simulating a disk failure before a real failure happens will ensure you are ready for when the real thing does happen.

Here is the high-level workflow for replacing a failed disk:

1. Find the failed OSD.
2. Take OSD out.
3. Stop the OSD daemon on the node.
4. Check Ceph's status.
5. Remove the OSD from the CRUSH map.
6. Delete the OSD authorization.
7. Remove the OSD from the storage cluster.
8. Unmount the filesystem on node.
9. Replace the failed drive.
10. Add the OSD back to the storage cluster.
11. Check Ceph's status.

2.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.
- A failed disk.

2.2. DISK FAILURES

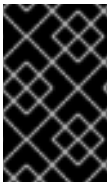
Ceph is designed for fault tolerance, which means Ceph can operate in a **degraded** state without losing data. Ceph can still operate even if a data storage drive fails. The **degraded** state means the extra copies of the data stored on other OSDs will backfill automatically to other OSDs in the storage cluster. When an OSD gets marked **down** this can mean the drive has failed.

When a drive fails, initially the OSD status will be **down**, but still **in** the storage cluster. Networking issues can also mark an OSD as **down** even if it is really **up**. First check for any network issues in the environment. If the networking checks out okay, then it is likely the OSD drive has failed.

Modern servers typically deploy with hot-swappable drives allowing you to pull a failed drive and replace it with a new one without bringing down the node. However, with Ceph you will also have to remove the software-defined part of the OSD.

2.2.1. Replacing a failed OSD disk

The general procedure for replacing an OSD involves removing the OSD from the storage cluster, replacing the drive and then recreating the OSD.



IMPORTANT

When replacing the BlueStore **block.db** disk that contains the BlueStore OSD's database partitions, Red Hat only supports the re-deploying of all OSDs using Ansible. A corrupt **block.db** files will impact all OSDs which are included in that **block.db** files.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A failed disk.

Procedure

1. Check storage cluster health:

```
# ceph health
```

2. Identify the OSD location in the CRUSH hierarchy:

```
# ceph osd tree | grep -i down
```

3. On the OSD node, try to start the OSD:

```
# systemctl start ceph-osd@$OSD_ID
```

If the command indicates that the OSD is already running, there might be a heartbeat or networking issue. If you cannot restart the OSD, then the drive might have failed.

NOTE

If the OSD is **down**, then the OSD will eventually get marked **out**. This is normal behavior for Ceph Storage. When the OSD gets marked **out**, other OSDs with copies of the failed OSD's data will begin backfilling to ensure that the required number of copies exist within the storage cluster. While the storage cluster is backfilling, the cluster will be in a **degraded** state.

4. For containerized deployments of Ceph, try to start the OSD container by referencing the drive associated with the OSD:

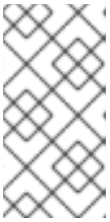
```
# systemctl start ceph-osd@$OSD_DRIVE
```

If the command indicates that the OSD is already running, there might be a heartbeat or networking issue. If you cannot restart the OSD, then the drive might have failed.

NOTE

The drive associated with the OSD can be determined by Mapping a container OSD ID to a drive.

5. Check the failed OSD's mount point:



NOTE

For containerized deployments of Ceph, if the OSD is down the container will be down and the OSD drive will be unmounted, so you cannot run **df** to check its mount point. Use another method to determine if the OSD drive has failed. For example, run **smartctl** on the drive from the container node.

```
# df -h
```

If you cannot restart the OSD, you can check the mount point. If the mount point no longer appears, then you can try remounting the OSD drive and restarting the OSD. If you cannot restore the mount point, then you might have a failed OSD drive.

Using the **smartctl** utility cab help determine if the drive is healthy. For example:

```
# yum install smartmontools
# smartctl -H /dev/$DRIVE
```

If the drive has failed, you will need to replace it.

6. Stop the OSD process:

```
# systemctl stop ceph-osd@$OSD_ID
```

   a. If using **FileStore**, then flush the journal to disk:

```
# ceph osd -i $$OSD_ID --flush-journal
```

7. For containerized deployments of Ceph, stop the OSD container by referencing the drive associated with the OSD:

```
# systemctl stop ceph-osd@$OSD_DRIVE
```

8. Remove the OSD out of the storage cluster:

```
# ceph osd out $OSD_ID
```

9. Ensure the failed OSD is backfilling:

```
# ceph -w
```

10. Remove the OSD from the CRUSH Map:

```
# ceph osd crush remove osd.$OSD_ID
```

NOTE

This step is only needed, if you are permanently removing the OSD and not redeploying it.

11. Remove the OSD's authentication keys:

```
# ceph auth del osd.$OSD_ID
```

12. Verify that the keys for the OSD are not listed:

```
# ceph auth list
```

13. Remove the OSD from the storage cluster:

```
# ceph osd rm osd.$OSD_ID
```

14. Unmount the failed drive path:

NOTE

For containerized deployments of Ceph, if the OSD is down the container will be down and the OSD drive will be unmounted. In this case there is nothing to unmount and this step can be skipped.

```
# umount /var/lib/ceph/osd/$CLUSTER_NAME-$OSD_ID
```

15. Replace the physical drive. Refer to the hardware vendor's documentation for the node. If the drive is hot swappable, simply replace the failed drive with a new drive. If the drive is NOT hot swappable and the node contains multiple OSDs, you MIGHT need to bring the node down to replace the physical drive. If you need to bring the node down temporarily, you might set the cluster to **noout** to prevent backfilling:

```
# ceph osd set noout
```

Once you replace the drive and you bring the node and its OSDs back online, remove the **noout** setting:

```
# ceph osd unset noout
```

Allow the new drive to appear under the **/dev/** directory and make a note of the drive path before proceeding further.

16. Find the OSD drive and format the disk.

17. Recreate the OSD:

    a. Using Ansible.

    b. Using the command-line interface.

18. Check the CRUSH hierarchy to ensure it is accurate:

```
# ceph osd tree
```

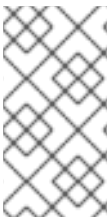
If you are not satisfied with the location of the OSD in the CRUSH hierarchy, you might move it with the **move** command:

```
# ceph osd crush move $BUCKET_TO_MOVE $BUCKET_TYPE=$PARENT_BUCKET
```

19. Verify the OSD is online.

2.2.2. Replacing an OSD drive while retaining the OSD ID

When replacing a failed OSD drive, you can keep the original OSD ID and CRUSH map entry.



NOTE

The **ceph-volume lvm** commands defaults to BlueStore for OSDs. To use FileStore OSDs, then use the **--filestore**, **--data** and **--journal** options.

See the Preparing the OSD Data and Journal Drives section for more details.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A failed disk.

Procedure

1. Destroy the OSD:

```
ceph osd destroy $OSD_ID --yes-i-really-mean-it
```

Example

```
$ ceph osd destroy 1 --yes-i-really-mean-it
```

2. Optionally, if the replacement disk was used previously, then you need to **zap** the disk:

```
ceph-volume lvm zap $DEVICE
```

Example

```
$ ceph-volume lvm zap /dev/sdb
```

3. Create the new OSD with the existing OSD ID:

```
ceph-volume lvm create --osd-id $OSD_ID --data $DEVICE
```

Example

```
$ ceph-volume lvm create --osd-id 1 --data /dev/sdb
```

2.3. SIMULATING A DISK FAILURE

There are two disk failure scenarios: hard and soft. A hard failure means replacing the disk. Soft failure might be an issue with the device driver or some other software component.

In the case of a soft failure, replacing the disk might not be needed. If replacing a disk, then steps need

to be followed to remove the failed disk and add the replacement disk to Ceph. In order to simulate a soft disk failure the best thing to do is delete the device. Choose a device and delete the device from the system.

```
echo 1 > /sys/block/$DEVICE/device/delete
```

Example

```
[root@ceph1 ~]# echo 1 > /sys/block/sdb/device/delete
```

In the Ceph OSD log, on the OSD node, Ceph detected the failure and started the recovery process automatically.

Example

```
[root@ceph1 ~]# tail -50 /var/log/ceph/ceph-osd.1.log
2017-02-02 12:15:27.490889 7f3e1fa3d800 -1 ^[[0;31m ** ERROR: unable to open OSD superblock
on /var/lib/ceph/osd/ceph-1: (5) Input/output error^[[0m
2017-02-02 12:34:17.777898 7fb7df1e7800  0 set uid:gid to 167:167 (ceph:ceph)
2017-02-02 12:34:17.777933 7fb7df1e7800  0 ceph version 10.2.3-17.el7cp
(ca9d57c0b140eb5cea9de7f7133260271e57490e), process ceph-osd, pid 1752
2017-02-02 12:34:17.788885 7fb7df1e7800  0 pidfile_write: ignore empty --pid-file
2017-02-02 12:34:17.870322 7fb7df1e7800  0 filestore(/var/lib/ceph/osd/ceph-1) backend xfs (magic
0x58465342)
2017-02-02 12:34:17.871028 7fb7df1e7800  0 genericfilestorebackend(/var/lib/ceph/osd/ceph-1)
detect_features: FIEMAP ioctl is disabled via 'filestore fiemap' config option
2017-02-02 12:34:17.871035 7fb7df1e7800  0 genericfilestorebackend(/var/lib/ceph/osd/ceph-1)
detect_features: SEEK_DATA/SEEK_HOLE is disabled via 'filestore seek data hole' config option
2017-02-02 12:34:17.871059 7fb7df1e7800  0 genericfilestorebackend(/var/lib/ceph/osd/ceph-1)
detect_features: splice is supported
2017-02-02 12:34:17.897839 7fb7df1e7800  0 genericfilestorebackend(/var/lib/ceph/osd/ceph-1)
detect_features: syncfs(2) syscall fully supported (by glibc and kernel)
2017-02-02 12:34:17.897985 7fb7df1e7800  0 xfsfilestorebackend(/var/lib/ceph/osd/ceph-1)
detect_feature: extsize is disabled by conf
2017-02-02 12:34:17.921162 7fb7df1e7800  1 leveldb: Recovering log #22
2017-02-02 12:34:17.947335 7fb7df1e7800  1 leveldb: Level-0 table #24: started
2017-02-02 12:34:18.001952 7fb7df1e7800  1 leveldb: Level-0 table #24: 810464 bytes OK
2017-02-02 12:34:18.044554 7fb7df1e7800  1 leveldb: Delete type=0 #22
2017-02-02 12:34:18.045383 7fb7df1e7800  1 leveldb: Delete type=3 #20
2017-02-02 12:34:18.058061 7fb7df1e7800  0 filestore(/var/lib/ceph/osd/ceph-1) mount: enabling
WRITEAHEAD journal mode: checkpoint is not enabled
2017-02-02 12:34:18.105482 7fb7df1e7800  1 journal _open /var/lib/ceph/osd/ceph-1/journal fd 18:
1073741824 bytes, block size 4096 bytes, directio = 1, aio = 1
2017-02-02 12:34:18.130293 7fb7df1e7800  1 journal _open /var/lib/ceph/osd/ceph-1/journal fd 18:
1073741824 bytes, block size 4096 bytes, directio = 1, aio = 1
2017-02-02 12:34:18.130992 7fb7df1e7800  1 filestore(/var/lib/ceph/osd/ceph-1) upgrade
2017-02-02 12:34:18.136547 7fb7df1e7800  0 <cls> cls/cephfs/cls_cephfs.cc:202: loading
cephfs_size_scan
2017-02-02 12:34:18.142863 7fb7df1e7800  0 <cls> cls/hello/cls_hello.cc:305: loading cls_hello
2017-02-02 12:34:18.255019 7fb7df1e7800  0 osd.1 51 crush map has features 2200130813952,
adjusting msgr requires for clients
2017-02-02 12:34:18.255041 7fb7df1e7800  0 osd.1 51 crush map has features 2200130813952 was
8705, adjusting msgr requires for mons
2017-02-02 12:34:18.255048 7fb7df1e7800  0 osd.1 51 crush map has features 2200130813952,
adjusting msgr requires for osds
```

```
2017-02-02 12:34:18.296256 7fb7df1e7800  0 osd.1 51 load_pgs
2017-02-02 12:34:18.561604 7fb7df1e7800  0 osd.1 51 load_pgs opened 152 pgs
2017-02-02 12:34:18.561648 7fb7df1e7800  0 osd.1 51 using 0 op queue with priority op cut off at 64.
2017-02-02 12:34:18.562603 7fb7df1e7800 -1 osd.1 51 log_to_monitors {default=true}
2017-02-02 12:34:18.650204 7fb7df1e7800  0 osd.1 51 done with init, starting boot process
2017-02-02 12:34:19.274937 7fb7b78ba700  0 -- 192.168.122.83:6801/1752 >>
192.168.122.81:6801/2620 pipe(0x7fb7ec4d1400 sd=127 :6801 s=0 pgs=0 cs=0 l=0
c=0x7fb7ec42e480).accept connect_seq 0 vs existing 0 state connecting
```

Looking at osd disk tree we also see the disk is offline.

```
[root@ceph1 ~]# ceph osd tree
ID WEIGHT  TYPE NAME      UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 0.28976 root default
-2 0.09659     host ceph3
 1 0.09659         osd.1       down 1.00000          1.00000
-3 0.09659     host ceph1
 2 0.09659         osd.2       up  1.00000          1.00000
-4 0.09659     host ceph2
 0 0.09659         osd.0       up  1.00000          1.00000
```

CHAPTER 3. HANDLING A NODE FAILURE

As a storage administrator, you might experience a whole node failing within the storage cluster, and handling a node failure is similar to handling a disk failure. With a node failure, instead of Ceph recovering PGs (placement groups) for only one disk, all PGs on the disks within that node must be recovered. Ceph will detect that the OSDs are all down and automatically start the recovery process, known as self-healing.

There are three node failure scenarios. Here is the high-level workflow for each scenario when replacing a node:

- Replacing the node, but using the root and Ceph OSD disks from the failed node.
  1. Disable backfilling.
  2. Replace the node, taking the disks from old node, and adding them to the new node.
  3. Enable backfilling.
- Replacing the node, reinstalling the operating system, and using the Ceph OSD disks from the failed node.
  1. Disable backfilling.
  2. Create a backup of the Ceph configuration.
  3. Replace the node and add the Ceph OSD disks from failed node.
     a. Configuring disks as JBOD.
  4. Install the operating system.
  5. Restore the Ceph configuration.
  6. Run **ceph-ansible**.
  7. Enable backfilling.
- Replacing the node, reinstalling the operating system, and using all new Ceph OSDs disks.
  1. Disable backfilling.
  2. Remove all OSDs on the failed node from the storage cluster.
  3. Create a backup of the Ceph configuration.
  4. Replace the node and add the Ceph OSD disks from failed node.
     a. Configuring disks as JBOD.
  5. Install the operating system.
  6. Run **ceph-ansible**.
  7. Enable backfilling.

3.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.
- A failed node.

3.2. CONSIDERATIONS BEFORE ADDING OR REMOVING A NODE

One of the outstanding features of Ceph is the ability to add or remove Ceph OSD nodes at run time. This means you can resize the storage cluster capacity or replace hardware without taking down the storage cluster. The ability to serve Ceph clients while the cluster is in a **degraded** state also has operational benefits, for example, you can add or remove or replace hardware during regular business hours, rather than working overtime or weekends. However, adding and removing Ceph OSD nodes can have a significant impact on performance, and you must consider the performance impact of adding, removing or replacing hardware on the storage cluster before you act.

From a capacity perspective, removing a node removes the OSDs contained within the node and effectively reduces the capacity of the storage cluster. Adding a node adds the OSDs contained within the node, and effectively expands the capacity of the storage cluster. Whether you are expanding or reducing the storage cluster capacity, adding or removing Ceph OSD nodes will induce backfilling as the cluster rebalances. During that rebalancing time period, Ceph uses additional resources which can impact storage cluster performance.

Imagine a storage cluster that contains Ceph nodes where each node has four OSDs. In a storage cluster of four nodes, with 16 OSDs, removing a node removes 4 OSDs and cuts capacity by 25%. In a storage cluster of three nodes, with 12 OSDs, adding a node adds 4 OSDs and increases capacity by 33%.

In a production Ceph storage cluster, a Ceph OSD node has a particular hardware configuration that facilitates a particular type of storage strategy. For more details, see Storage Strategies guide for Red Hat Ceph Storage 3.

Since a Ceph OSD node is part of a CRUSH hierarchy, the performance impact of adding or removing a node typically affects the performance of pools that use that CRUSH hierarchy, that is, the CRUSH ruleset.

3.3. PERFORMANCE CONSIDERATIONS

The following factors typically have an impact on storage cluster's performance when adding or removing Ceph OSD nodes:

Current Client Load on Affected Pools:

Ceph clients place load on the I/O interface to Ceph; namely, load on a pool. A pool maps to a CRUSH ruleset. The underlying CRUSH hierarchy allows Ceph to place data across failure domains. If the underlying Ceph OSD node involves a pool under high client loads, the client load may have a significant impact on recovery time and impact performance. More specifically, since write operations require data replication for durability, write-intensive client loads will increase the time for the storage cluster to recover.

Capacity Added or Removed:

Generally, the capacity you are adding or removing as a percentage of the overall cluster will have an impact on the storage cluster's time to recover. Additionally, the storage density of the node you add or remove may have an impact on the time to recover for example, a node with 36 OSDs will typically take longer to recover compared to a node with 12 OSDs. When removing nodes, you MUST ensure that you have sufficient spare capacity so that you will not reach the **full ratio** or **near full ratio**. If the storage cluster reaches the **full ratio**, Ceph will suspend write operations to prevent data loss.

Pools and CRUSH Ruleset:

A Ceph OSD node maps to at least one Ceph CRUSH hierarchy, and the hierarchy maps to at least one pool. Each pool that uses the CRUSH hierarchy (ruleset) where you add or remove a Ceph OSD node will experience a performance impact.

Pool Type and Durability:

Replication pools tend to use more network bandwidth to replicate deep copies of the data, whereas erasure coded pools tend to use more CPU to calculate **k+m** coding chunks. The more copies of the data, for example, the size or the more **k+m** chunks, the longer it will take for the storage cluster to recover.

Total Throughput Characteristics:

Drives, controllers and network interface cards all have throughput characteristics that may impact the recovery time. Generally, nodes with higher throughput characteristics, for example, 10 Gbps and SSDs will recover faster than nodes with lower throughput characteristics, for example, 1 Gbps and SATA drives.

3.4. RECOMMENDATIONS FOR ADDING OR REMOVING NODES

The failure of a node may preclude removing one OSD at a time before changing the node. Circumstances can allow you to reduce a negative performance impact when adding or removing Ceph OSD nodes, Red Hat recommends adding or removing one OSD at a time within a node and allowing the cluster to recover before proceeding to the next OSD. For details on removing an OSD:

- Using Ansible.
- Using the command-line interface.

When adding a Ceph node, Red hat also recommends adding one OSD at a time. For details on adding an OSD:

- Using Ansible.
- Using the command-line interface.

When adding or removing Ceph OSD nodes, consider that other ongoing processes will have an impact on performance too. To reduce the impact on client I/O, Red Hat recommends the following:

Calculate capacity:

Before removing a Ceph OSD node, ensure that the storage cluster can backfill the contents of all its OSDs **WITHOUT** reaching the **full ratio**. Reaching the **full ratio** will cause the cluster to refuse write operations.

Temporarily Disable Scrubbing:

Scrubbing is essential to ensuring the durability of the storage cluster's data; however, it is resource intensive. Before adding or removing a Ceph OSD node, disable scrubbing and deep scrubbing and let the current scrubbing operations complete before proceeding, for example:

```
ceph osd set noscrub
ceph osd set nodeep-scrub
```

Once you have added or removed a Ceph OSD node and the storage cluster has returned to an **active+clean** state, unset the **noscrub** and **nodeep-scrub** settings.

Limit Backfill and Recovery:

If you have reasonable data durability, for example, **osd pool default size = 3** and **osd pool default min size = 2**, there is nothing wrong with operating in a **degraded** state. You can tune the storage cluster for the fastest possible recovery time, but this will impact Ceph client I/O performance significantly. To maintain the highest Ceph client I/O performance, limit the backfill and recovery operations and allow them to take longer, for example:

```
osd_max_backfills = 1
osd_recovery_max_active = 1
osd_recovery_op_priority = 1
```

You can also set sleep and delay parameters such as **osd_recovery_sleep**.

Finally, if you are expanding the size of the storage cluster, you may need to increase the number of placement groups. If you determine that you need to expand the number of placement groups, Red Hat recommends making incremental increases in the number of placement groups. Increasing the number of placement groups by a significant number will cause performance to degrade considerably.

3.5. ADDING A CEPH OSD NODE

To expand the capacity of the Red Hat Ceph Storage cluster, add an OSD node.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A provisioned node with a network connection.
- Installation of Red Hat Enterprise Linux 7 or Ubuntu 16.04.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the Installation Guide for Red Hat Enterprise Linux or Ubuntu.

Procedure

1. Verify that other nodes in the storage cluster can reach the new node by its short host name.
2. Temporarily disable scrubbing:

```
[root@monitor ~]# ceph osd set noscrub
[root@monitor ~]# ceph osd set nodeep-scrub
```

3. Limit the back-fill and recovery features:

Syntax

```
ceph tell $DAEMON_TYPE.* injectargs --$OPTION_NAME $VALUE [--$OPTION_NAME
$VALUE]
```

Example

```
[root@monitor ~]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active
1 --osd-recovery-op-priority 1
```

4. Add the new node to the CRUSH Map:

   Syntax

   ```
   ceph osd crush add-bucket $BUCKET_NAME $BUCKET_TYPE
   ```

   Example

   ```
   [root@monitor ~]# ceph osd crush add-bucket node2 host
   ```

5. Add an OSD for each disk on the node to the storage cluster.

   - Using Ansible.
   - Using the command-line interface.

   IMPORTANT

   When adding an OSD node to a Red Hat Ceph Storage cluster Red Hat recommends adding one OSD at a time within the node and allowing the cluster to recover to an **active+clean** state before proceeding to the next OSD.

Additional Resources

- See the Setting a Specific Configuration Setting at Runtime section in the Red Hat Ceph Storage Configuration Guide for more details.
- See the Add a Bucket and Move a Bucket sections in the Red Hat Ceph Storage Storage Strategies Guide for details on placing the node at an appropriate location in the CRUSH hierarchy,.

3.6. REMOVING A CEPH OSD NODE

To reduce the capacity of a storage cluster remove an OSD node.

WARNING

Before removing a Ceph OSD node, ensure that the storage cluster can backfill the contents of all OSDs WITHOUT reaching the full ratio. Reaching the full ratio will cause the cluster to refuse write operations.

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. Check storage cluster's capacity:

```
[root@monitor ~]# ceph df
[root@monitor ~]# rados df
[root@monitor ~]# ceph osd df
```

2. Temporarily disable scrubbing:

```
[root@monitor ~]# ceph osd set noscrub
[root@monitor ~]# ceph osd set nodeep-scrub
```

3. Limit the back-fill and recovery features:

Syntax

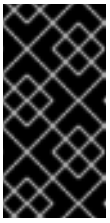
```
ceph tell $DAEMON_TYPE.* injectargs --$OPTION_NAME $VALUE [--$OPTION_NAME
$VALUE]
```

Example

```
[root@monitor ~]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active
1 --osd-recovery-op-priority 1
```

4. Remove each OSD on the node from the storage cluster:

   - Using Ansible.
   - Using the command-line interface.



IMPORTANT

When removing an OSD node from the storage cluster, Red Hat recommends removing one OSD at a time within the node and allowing the cluster to recover to an **active+clean** state before proceeding to the next OSD.

   a. After removing an OSD check to verify the storage cluster is not getting to the near-full ratio:

```
[root@monitor ~]# ceph -s
[root@monitor ~]# ceph df
```

   b. Repeat this step until all OSDs on the node are removed from the storage cluster.

5. Once all OSDs are removed, remove the host bucket from the CRUSH map:

Syntax

```
ceph osd crush rm $BUCKET_NAME
```

Example

```
[root@monitor ~]# ceph osd crush rm node2
```

Additional Resources

- * See the Setting a Specific Configuration Setting at Runtime section in the Red Hat Ceph Storage Configuration Guide for more details.

3.7. SIMULATING A NODE FAILURE

To simulate hard node failure power-off the node and reinstall the operating system.

Prerequisites

- A healthy running Red Hat Ceph Storage cluster.

Procedure

1. Check storage capacity to understand what removing node means to storage cluster:

```
# ceph df
# rados df
# ceph osd df
```

2. Optionally, disable recovery and backfilling:

```
# ceph osd set noout
# ceph osd set noscrub
# ceph osd set nodeep-scrub
```

3. Shutdown the node.

4. If the host name will change, then remove the node from CRUSH map:

```
[root@ceph1 ~]# ceph osd crush rm ceph3
```

5. Check status of cluster:

```
[root@ceph1 ~]# ceph -s
```

6. Reinstall the operating system on the node.

7. Add an Ansible user and SSH keys:

```
[root@ceph3 ~]# useradd ansible
[root@ceph3 ~]# passwd ansible
[root@ceph3 ~]# cat << EOF > /etc/sudoers.d/ansible
ansible ALL = (root) NOPASSWD:ALL
Defaults:ansible !requiretty
EOF
[root@ceph3 ~]# su - ansible
[ansible@ceph3 ~]# ssh-keygen
```

8. From the administration node, copy the SSH keys for **ansible** user:

```
[ansible@admin ~]$ ssh-copy-id ceph3
```

9. From the administration node, re-run the Ansible playbook:

```
[ansible@admin ~]$ cd /usr/share/ceph-ansible
[ansible@admin ~]$ ansible-playbook site.yml
```

Example Output

```
PLAY RECAP ********************************************************************
ceph1                      : ok=368  changed=2    unreachable=0    failed=0
ceph2                      : ok=284  changed=0    unreachable=0    failed=0
ceph3                      : ok=284  changed=15   unreachable=0    failed=0
```

10. Optionally, enable recovery and backfilling:

```
[root@ceph3 ~]# ceph osd unset noout
[root@ceph3 ~]# ceph osd unset noscrub
[root@ceph3 ~]# ceph osd unset nodeep-scrub
```

11. Check Ceph's health:

```
[root@ceph3 ~]# ceph -s
    cluster 1e0c9c34-901d-4b46-8001-0d1f93ca5f4d
     health HEALTH_OK
     monmap e1: 3 mons at
{ceph1=192.168.122.81:6789/0,ceph2=192.168.122.82:6789/0,ceph3=192.168.122.83:6789/0}

            election epoch 36, quorum 0,1,2 ceph1,ceph2,ceph3
     osdmap e95: 3 osds: 3 up, 3 in
            flags sortbitwise
      pgmap v1190: 152 pgs, 12 pools, 1024 MB data, 441 objects
            3197 MB used, 293 GB / 296 GB avail
                 152 active+clean
```

Additional Resources

- For more information on installing Red Hat Ceph Storage:
  - Red Hat Enterprise Linux
  - Ubuntu

CHAPTER 4. HANDLING A DATA CENTER FAILURE

Red Hat Ceph Storage can withstand catastrophic failures to the infrastructure, such as losing one of three data centers in a stretch cluster. For the standard object store use case, configuring all three data centers can be done independently with replication set up between them. In this scenario, the cluster configuration in each of the data centers might be different, reflecting the local capabilities and dependencies.

A logical structure of the placement hierarchy should be considered. A proper CRUSH map can be used, reflecting the hierarchical structure of the failure domains within the infrastructure. Using logical hierarchical definitions improves the reliability of the storage cluster, versus using the standard hierarchical definitions. Failure domains are defined in the CRUSH map. The default CRUSH map contains all nodes in a flat hierarchy.

In three data center environment example, with a stretch cluster, the placement of nodes should be managed in a way that one data center can go down, but the storage cluster stays up and running. Consider which failure domain a node resides in when using 3-way replication for the data, in the case of an outage of one data center, it is possible that some data can be left with one copy. When this scenario happens, there are two options:

- Leave the data in read-only status with the standard settings.
- Live with only one copy for the duration of the outage.

With the standard settings, and because of the randomness of data placement across the nodes, not all the data will be affected, but some data can have only one copy and the storage cluster would revert to read-only mode.

In the example below the resulting map is derived from the initial setup of the cluster with 6 OSD nodes. In this example all nodes have only one disk and hence one OSD. All of the nodes are arranged under the default *root*, that is the standard *root* of the hierarchy tree. Because there is a weight assigned to two of the OSDs, these OSDs receive fewer chunks of data than the other OSDs. These nodes were introduced later with bigger disks than the initial OSD disks. This does not affect the data placement to withstand a failure of a group of nodes.

Standard CRUSH map

```
$ sudo ceph osd tree
ID WEIGHT  TYPE NAME          UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 0.33554 root default
-2 0.04779     host ceph-node3
 0 0.04779         osd.0           up  1.00000          1.00000
-3 0.04779     host ceph-node2
 1 0.04779         osd.1           up  1.00000          1.00000
-4 0.04779     host ceph-node1
 2 0.04779         osd.2           up  1.00000          1.00000
-5 0.04779     host ceph-node4
 3 0.04779         osd.3           up  1.00000          1.00000
-6 0.07219     host ceph-node6
 4 0.07219         osd.4           up  0.79999          1.00000
-7 0.07219     host ceph-node5
 5 0.07219         osd.5           up  0.79999          1.00000
```

Using logical hierarchical definitions to group the nodes into same data center, can achieve data placement maturity. Possible definition types of *root*, *datacenter*, *rack*, *row* and *host* allow the reflection of the failure domains for the three data center stretch cluster:

- Nodes ceph-node1 and ceph-node2 reside in data center 1 (DC1)
- Nodes ceph-node3 and ceph-node5 reside in data center 2 (DC2)
- Nodes ceph-node4 and ceph-node6 reside in data center 3 (DC3)
- All data centers belong to the same structure (allDC)

Since all OSDs in a host belong to the host definition there is no change needed. All the other assignments can be adjusted during runtime of the storage cluster by:

- Defining the *bucket* structure with the following commands:

```
ceph osd crush add-bucket allDC root
ceph osd crush add-bucket DC1 datacenter
ceph osd crush add-bucket DC2 datacenter
ceph osd crush add-bucket DC3 datacenter
```

- Moving the nodes into the appropriate place within this structure by modifying the CRUSH map:

```
ceph osd crush move DC1 root=allDC
ceph osd crush move DC2 root=allDC
ceph osd crush move DC3 root=allDC
ceph osd crush move ceph-node1 datacenter=DC1
ceph osd crush move ceph-node2 datacenter=DC1
ceph osd crush move ceph-node3 datacenter=DC2
ceph osd crush move ceph-node5 datacenter=DC2
ceph osd crush move ceph-node4 datacenter=DC3
ceph osd crush move ceph-node6 datacenter=DC3
```

Within this structure any new hosts can be added too, as well as new disks. By placing the OSDs at the right place in the hierarchy the CRUSH algorithm is changed to place redundant pieces into different failure domains within the structure.

The above example results in the following:

```
$ sudo ceph osd tree
ID  WEIGHT  TYPE NAME             UP/DOWN REWEIGHT PRIMARY-AFFINITY
 -8 6.00000 root allDC
 -9 2.00000     datacenter DC1
 -4 1.00000         host ceph-node1
  2 1.00000             osd.2          up  1.00000          1.00000
 -3 1.00000         host ceph-node2
  1 1.00000             osd.1          up  1.00000          1.00000
-10 2.00000     datacenter DC2
 -2 1.00000         host ceph-node3
  0 1.00000             osd.0          up  1.00000          1.00000
 -7 1.00000         host ceph-node5
  5 1.00000             osd.5          up  0.79999          1.00000
-11 2.00000     datacenter DC3
 -6 1.00000         host ceph-node6
  4 1.00000             osd.4          up  0.79999          1.00000
 -5 1.00000         host ceph-node4
  3 1.00000             osd.3          up  1.00000          1.00000
 -1       0 root default
```

The listing from above shows the resulting CRUSH map by displaying the osd tree. Easy to see is now how the hosts belong to a data center and all data centers belong to the same top level structure but clearly distinguishing between locations.

NOTE

Placing the data in the proper locations according to the map works only properly within the healthy cluster. Misplacement might happen under circumstances, when some OSDs not available. Those misplacements will be corrected automatically once it's possible to do so.

Additional Resources

- See the CRUSH administration chapter in the Red Hat Ceph Storage Storage Strategies Guide for more information.