Red Hat Ceph Storage 4

Operations Guide

Operational tasks for Red Hat Ceph Storage

Last Updated: 2024-01-19

Red Hat Ceph Storage 4 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. Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message .

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. You can manage the storage cluster size by using Ceph Ansible, or by using the command-line interface (CLI).

1.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor and OSD nodes.

1.2. CEPH MONITORS

Ceph Monitors are lightweight processes that maintain a master copy of the storage cluster map. All Ceph clients contact a Ceph monitor and retrieve the current copy of the storage 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 storage cluster. Due to the nature of Paxos, Ceph requires a majority of monitors running to establish a quorum, thus establishing consensus.

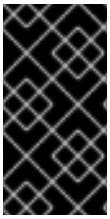
IMPORTANT

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 Ceph 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 Ceph Monitors are lightweight, 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 Ceph 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

Before you prepare a new Ceph Monitor node for deployment, review the *Requirements for Installing Red Hat Ceph Storage* chapter in the *Red Hat Ceph Storage Installation Guide*.

IMPORTANT

Deploy each new Ceph Monitor on a separate node, and all Ceph Monitor nodes in the storage cluster must run on the same hardware.

Prerequisites

- Network connectivity.
- Root-level access to the new node.

Procedure

1. Add the new node to the server rack.
2. Connect the new node to the network.
3. Install the latest version of Red Hat Enterprise Linux 7 or Red Hat Enterprise Linux 8.
   a. For Red Hat Enterprise Linux 7, install **ntp** and configure a reliable time source:

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

   b. For Red Hat Enterprise Linux 8, install **chrony** and configure a reliable time source:

   ```
   [root@mon ~]# dnf install chrony
   ```

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

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

Additional Resources

- For more information about **chrony**, refer to *Red Hat Enterprise Linux 8 Configuring basic system settings*.

1.2.2. Adding a Ceph Monitor using Ansible

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

Prerequisites

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

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:

   ```
   [root@admin ~]# ansible all -m ping
   ```

3. Change directory to the Ansible configuration directory:

   ```
   [root@admin ~]# cd /usr/share/ceph-ansible
   ```

4. You can add a Ceph Monitor using either of the following steps:

   - For both **bare-metal** and **containers** deployments, run the **infrastructure-playbook**:

     ```
     [root@admin ceph-ansible]# ansible-playbook -vvvv -i hosts infrastructure-playbooks/add-mon.yml
     ```

   - As the ansible user, run either the **site** playbook or the **site-container** playbook:

     - **Bare-metal** deployments:

       Example

       ```
       [ansible@admin ceph-ansible]$ ansible-playbook -vvvv -i hosts site.yml --limit mons
       ```

     - **Container** deployments:

       Example

       ```
       [ansible@admin ceph-ansible]$ ansible-playbook -vvvv -i hosts site-container.yml --limit mons
       ```

       After the Ansible playbook has finished running, the new Ceph Monitor nodes appear in the storage cluster.

5. Update the configuration file:

a. **Bare-metal** deployments:

Example

```
[user@admin ceph-ansible]$ ansible-playbook -vvvv -i hosts site.yml --tags
ceph_update_config
```

b. **Container** deployments:

Example

```
[user@admin ceph-ansible]$ ansible-playbook -vvvv -i hosts site-container.yml --tags
ceph_update_config
```

Additional Resources

- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

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

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

IMPORTANT

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

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to a running Ceph Monitor node and to the new monitor nodes.

Procedure

1. Add the Red Hat Ceph Storage 4 monitor repository.

   Red Hat Enterprise Linux 7

   ```
   [root@mon ~]# subscription-manager repos --enable=rhel-7-server-rhceph-4-mon-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   [root@mon ~]# subscription-manager repos --enable=rhceph-4-mon-for-rhel-8-x86_64-rpms
   ```

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

   Red Hat Enterprise Linux 7

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

Red Hat Enterprise Linux 8

```
[root@mon ~]# dnf install ceph-mon
```

3. Edit the **mon_host** settings list in the **[mon]** section of the Ceph configuration file on a running node in the storage cluster.

   a. Add the IP address of the new Ceph Monitor node to the **mon_host** settings list:

   Syntax

   ```
   [mon]
   mon_host = MONITOR_IP : PORT MONITOR_IP : PORT ... NEW_MONITOR_IP :
   PORT
   ```

   Instead of adding the new Ceph Monitor's IP address to the [mon] section of the Ceph configuration file, you can create a specific section in the file for the new monitor nodes:

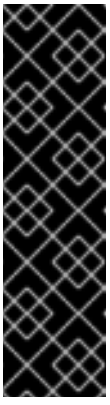
   Syntax

   ```
   [mon.MONITOR_ID]
   host = MONITOR_ID
   mon_addr = MONITOR_IP
   ```

   NOTE

   The **mon_host** settings list is a list of DNS-resolvable host names or IP addresses, separated by "," or ";" or " ". This list ensures that the storage cluster identifies the new Monitor node during a start or restart.

   IMPORTANT

   The **mon_initial_members** setting lists the initial quorum group of Ceph Monitor nodes. If one member of that group fails, another node in that group becomes the initial monitor node. To ensure high availability for production storage clusters, list at least three monitor nodes in the **mon_initial_members** and **mon_host** sections of the Ceph configuration file. This prevents the storage cluster from locking up if the initial monitor node fails. If the Monitor nodes you are adding are replacing monitors that were part of **mon_initial_members** and **mon_host**, add the new monitors to both sections as well.

4. To make the monitors part of the initial quorum group, 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
```

5. 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@mon ~]# scp /etc/ceph/ceph.conf node4:/etc/ceph
   ```

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

   Syntax

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

   Example

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

7. Create temporary directories on a running Ceph Monitor node and on the new monitor nodes, and keep the files needed for this procedure in those directories. The temporary directory on each node should be different from the node's default directory. It can be removed after all the steps are completed:

   Syntax

   ```
   mkdir TEMP_DIRECTORY_PATH_NAME
   ```

   Example

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

8. Copy the admin key from a running Ceph Monitor node to the new Ceph 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@mon ~]# scp /etc/ceph/ceph.client.admin.keyring node4:/etc/ceph
```

9. From a running Ceph Monitor node, retrieve the monitor keyring:

   Syntax

   ```
   ceph auth get mon. -o TEMP_DIRECTORY_PATH_NAME/KEY_FILE_NAME
   ```

   Example

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

10. From a running Ceph Monitor node, retrieve the monitor map:

    Syntax

    ```
    ceph mon getmap -o TEMP_DIRECTORY_PATH_NAME/MONITOR_MAP_FILE
    ```

    Example

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

11. Copy the collected Ceph Monitor data to the new Ceph Monitor nodes:

    Syntax

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

    Example

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

12. Prepare the data directory for the new monitors from the data you collected earlier. Specify the path to the monitor map to retrieve quorum information from the monitors, along with their `fsid`s. Specify a path to the monitor keyring:

    Syntax

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

    Example

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

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

    Syntax

    ```
    ```

```
echo "CLUSTER=CUSTOM_CLUSTER_NAME" >> /etc/sysconfig/ceph
```

Example

```
[root@mon ~]# echo "CLUSTER=example" >> /etc/sysconfig/ceph
```

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

Syntax

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

Example

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

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

Additional Resources

- See the *Enabling the Red Hat Ceph Storage Repositories* section in the *Red Hat Ceph Storage Installation Guide*.

1.2.4. Configuring monitor election strategy

The monitor election strategy identifies the net splits and handles failures. You can configure the election monitor strategy in three different modes:

1. **classic** - This is the default mode in which the lowest ranked monitor is voted based on the elector module between the two sites.

2. **disallow** - This mode lets you mark monitors as disallowed, in which case they will participate in the quorum and serve clients, but cannot be an elected leader. This lets you add monitors to a list of disallowed leaders. If a monitor is in the disallowed list, it will always defer to another monitor.

3. **connectivity** - This mode is mainly used to resolve network discrepancies. It evaluates

connection scores provided by each monitor for its peers and elects the most connected and reliable monitor to be the leader. This mode is designed to handle net splits, which may happen if your cluster is stretched across multiple data centers or otherwise susceptible. This mode incorporates connection score ratings and elects the monitor with the best score.

Red Hat recommends you to stay in the **classic** mode unless you require features in the other modes.

Before constructing the cluster, change the **election_strategy** to **classic**, **disallow**, or **connectivity** in the following command:

Syntax

```
ceph mon set election_strategy {classic|disallow|connectivity}
```

1.2.5. 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. Check if the monitor is **ok-to-stop**:

   #### Syntax

   ```
   ceph mon ok-to-stop MONITOR_ID
   ```

   #### Example

   ```
   [root@mon ~]# ceph mon ok-to-stop node03
   ```

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

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

3. For **bare-metal** and **containers** deployments, run the **shrink-mon.yml** Ansible playbook:

   #### Syntax

   ```
   ansible-playbook infrastructure-playbooks/shrink-mon.yml -e mon_to_kill=NODE_NAME -u ANSIBLE_USER_NAME -i hosts
   ```

   Replace:

   - ***NODE_NAME*** with the short host name of the Ceph Monitor node. You can remove only one Ceph Monitor each time the playbook runs.
   - ***ANSIBLE_USER_NAME*** with the name of the Ansible user

Example

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/shrink-mon.yml -e mon_to_kill=node03 -u user -i hosts
```

4. Manually remove the corresponding entry from the ansible inventory host **/etc/ansible/hosts**.

5. Run the **ceph-ansible** playbook.

   a. **Bare-metal deployments**:

      Example

      ```
      [user@admin ceph-ansible]$ ansible-playbook site.yml --tags ceph_update_config -i hosts
      ```

   b. **Container deployments**:

      Example

      ```
      [user@admin ceph-ansible]$ ansible-playbook site-container.yml --tags ceph_update_config -i hosts
      ```

6. Ensure that the Ceph Monitor has been successfully removed:

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

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

1.2.6. 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.
- Root-level access to the monitor node.

Procedure

1. Check if the monitor is **ok-to-stop**:

   Syntax

   ```
   ceph mon ok-to-stop HOSTNAME
   ```

Example

```
[root@mon ~]# ceph mon ok-to-stop node03
```

2. Stop the Ceph Monitor service:

   Syntax

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

   Example

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

3. Remove the Ceph Monitor from the storage cluster:

   Syntax

   ```
   ceph mon remove MONITOR_ID
   ```

   Example

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

4. Remove the Ceph Monitor entry from the Ceph configuration file. The default location for the configuration file is **/etc/ceph/ceph.conf**.

5. 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@mon ~]# scp /etc/ceph/ceph.conf root@node3:/etc/ceph/
   ```

6. For a **Containers** deployment, disable and remove the Ceph Monitor service:

   a. Disable the Ceph Monitor service:

      Syntax

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

      Example

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

   b. Remove the service from **systemd**:

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

c. Reload the **systemd** manager configuration:

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

d. Reset the state of the failed Ceph Monitor node:

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

7. Optional: Archive the Ceph Monitor data:

Syntax

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

Example

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

8. Optional: Delete the Ceph Monitor data:

Syntax

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

Example

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

1.2.7. Removing a Ceph Monitor from an unhealthy storage cluster

You can remove a **ceph-mon** daemon from an unhealthy storage cluster that has placement groups persistently not in **active + clean** state.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor node.
- At least one running Ceph Monitor node.

Procedure

1. Log into a surviving Ceph Monitor node:

Syntax

```
ssh root@MONITOR_NODE_NAME
```

Example

```
[root@admin ~]# ssh root@mon2
```

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

Syntax

```
systemctl stop ceph-mon@MONITOR_ID
ceph-mon -i SHORT_HOSTNAME --extract-monmap TEMP_PATH
```

Example

```
[root@mon2 ~]# systemctl stop ceph-mon@mon1
[root@mon2 ~]# ceph-mon -i mon1 --extract-monmap /tmp/monmap
```

3. Remove the non-surviving Ceph Monitor(s):

Syntax

```
monmaptool TEMPORARY_PATH --rm _MONITOR_ID
```

Example

```
[root@mon2 ~]# monmaptool /tmp/monmap --rm mon1
```

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

Syntax

```
ceph-mon -i SHORT_HOSTNAME --inject-monmap TEMP_PATH
```

Example

```
[root@mon2 ~]# ceph-mon -i mon2 --inject-monmap /tmp/monmap
```

5. Start only the surviving monitors, and verify that the monitors form a quorum:

Example

```
[root@mon2 ~]# ceph -s
```

6. Optional: Archive the removed Ceph Monitor's data directory in **/var/lib/ceph/mon** directory.

1.3. CEPH MANAGERS

The Ceph Manager daemon (**ceph-mgr**) runs alongside monitor daemons, to provide additional monitoring and interfaces to external monitoring and management systems. The **ceph-mgr** daemon is required for normal operations. By default, the Ceph Manager daemon requires no additional

configuration, beyond ensuring it is running. If there is no mgr daemon running, you can see a health warning to that effect, and some of the other information in the output of **ceph status** is missing or stale until a Ceph Manager is started.

1.3.1. Adding a Ceph Manager using Ansible

Usually, the Ansible automation utility installs the Ceph Manager daemon (**ceph-mgr**) when you deploy the Red Hat Ceph Storage cluster. If the Ceph Manager service or the daemon is down, you can redeploy the **ceph-mgr** daemon using Ansible. You can remove the manager daemon and add a new or an existing node to deploy the Ceph Manager daemon. Red Hat recommends colocating the Ceph Manager and Ceph Monitor daemons on the same node.

Prerequisites

- A running Red Hat Ceph Storage cluster deployed by Ansible.
- **Root** or **sudo** access to an Ansible administration node.
- New or existing nodes to deploy the Ceph Manager daemons.

Procedure

1. Log in to the Ansible administration node.
2. Navigate to the **/usr/share/ceph-ansible/** directory:

   Example

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

3. As **root** or with **sudo** access, open and edit the **/usr/share/ceph-ansible/hosts** inventory file and add the Ceph Manager node under the **[mgrs]** section:

   Syntax

   ```
   [mgrs]
   CEPH_MANAGER_NODE_NAME
   CEPH_MANAGER_NODE_NAME
   ```

   Replace *CEPH_MANAGER_NODE_NAME* with the host name of the node where you want to install the Ceph Manager daemon.

4. As the **ansible** user, run the Ansible playbook:

   - **Bare-metal** deployments:

     ```
     [ansible@admin ceph-ansible]$ ansible-playbook site.yml --limit mgrs -i hosts
     ```

   - **Container** deployments:

     ```
     [ansible@admin ceph-ansible]$ ansible-playbook site-container.yml --limit mgrs -i hosts
     ```

     After the Ansible playbook has finished running, the new Ceph Manager daemons node appears in the storage cluster.

Verification

- On the monitor node, check the status of the storage cluster:

  Syntax

  ```
  ceph -s
  ```

  Example

  ```
  [root@ceph-2 ~]# ceph -s

  mgr: ceph-3(active, since 2h), standbys: ceph-1, ceph-2
  ```

Additional Resources

- See the *Manually installing Ceph Manager* section in the *Red Hat Ceph Storage Installation Guide* for more details on many adding a Ceph Manager daemon to a bare-metal storage cluster.
- See the *Removing a Ceph Manager using Ansible* section in the *Red Hat Ceph Storage Operations Guide* for more details.

1.3.2. Removing a Ceph Manager using Ansible

You can use **shrink-mgr** playbook to remove the Ceph Manager daemons. This playbook removes a Ceph manager from your cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster deployed by Ansible.
- **Root** or **sudo** access to an Ansible administration node.
- Admin access to the Ansible administration node.

Procedure

1. As an admin user, log in to the Ansible administration node.
2. Navigate to the **/usr/share/ceph-ansible/** directory.

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

3. For **bare-metal** and **containers** deployments, run the **shrink-mgr.yml** Ansible playbook:

   Syntax

   ```
   ansible-playbook infrastructure-playbooks/shrink-mgr.yml -e mgr_to_kill=NODE_NAME -u
   ANSIBLE_USER_NAME -i hosts
   ```

   Replace:

- ***NODE_NAME*** with the short host name of the Ceph Manager node. You can remove only one Ceph Manager each time the playbook runs.
- ***ANSIBLE_USER_NAME*** with the name of the Ansible user

Example

```
[admin@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/shrink-mgr.yml -e mgr_to_kill=ceph-2 -u admin -i hosts
```

4. As a **root** user, edit the **/usr/share/ceph-ansible/hosts** inventory file and remove the Ceph Manager node under the **[mgrs]** section:

Syntax

```
[mgrs]
CEPH_MANAGER_NODE_NAME
CEPH_MANAGER_NODE_NAME
```

Example

```
[mgrs]
ceph-1
ceph-3
```

In this example, **ceph-2** was removed from the **[mgrs]** list.

Verification

- On the monitor node, check the status of the storage cluster:

Syntax

```
ceph -s
```

Example

```
[root@ceph-2 ~]# ceph -s

mgr: ceph-3(active, since 112s), standbys: ceph-1
```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- See the *Configuring Ansible's inventory location* section in the *Red Hat Ceph Storage Installation Guide* for more details on the Ansible inventory configuration.

1.4. CEPH MDSS

The Ceph Metadata Server (MDS) node runs the MDS daemon (ceph-mds), which manages metadata

related to files stored on the Ceph File System (CephFS). The MDS provides POSIX-compliant, shared file-system metadata management, including ownership, time stamps, and mode. The MDS uses RADOS (Reliable Autonomic Distributed Object Storage) to store metadata.

The MDS enables CephFS to interact with the Ceph Object Store, mapping an inode to an object and the location where Ceph stores the data within a tree. Clients accessing a CephFS file system first make a request to an MDS, which provides the information needed to get file content from the correct OSDs.

1.4.1. Adding a Ceph MDS using Ansible

Use the Ansible playbook to add a Ceph Metadata Server (MDS).

Prerequisites

- A running Red Hat Ceph Storage cluster deployed by Ansible.
- **Root** or **sudo** access to an Ansible administration node.
- New or existing servers that can be provisioned as MDS nodes.

Procedure

1. Log in to the Ansible administration node
2. Change to the **/usr/share/ceph-ansible** directory:

   Example

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

3. As **root** or with **sudo** access, open and edit the **/usr/share/ceph-ansible/hosts** inventory file and add the MDS node under the **[mdss]** section:

   Syntax

   ```
   [mdss]
   MDS_NODE_NAME
   NEW_MDS_NODE_NAME
   ```

   Replace *NEW_MDS_NODE_NAME* with the host name of the node where you want to install the MDS server.

   Alternatively, you can colocate the MDS daemon with the OSD daemon on one node by adding the same node under the **[osds]** and **[mdss]** sections.

   Example

   ```
   [mdss]
   node01
   node03
   ```

4. As the **ansible** user, run the Ansible playbook to provision the MDS node:

   - **Bare-metal** deployments:

```
[ansible@admin ceph-ansible]$ ansible-playbook site.yml --limit mdss -i hosts
```

- **Container** deployments:

```
[ansible@admin ceph-ansible]$ ansible-playbook site-container.yml --limit mdss -i hosts
```

After the Ansible playbook has finished running, the new Ceph MDS node appears in the storage cluster.

Verification

- Check the status of the MDS daemons:

  Syntax

```
ceph fs dump
```

  Example

```
[ansible@admin ceph-ansible]$ ceph fs dump

[mds.node01 {0:115304} state up:active seq 5 addr
[v2:172.25.250.10:6800/695510951,v1:172.25.250.10:6801/695510951]]

Standby daemons:
[mds.node03 {-1:144437} state up:standby seq 2 addr
[v2:172.25.250.11:6800/172950087,v1:172.25.250.11:6801/172950087]]
```

- Alternatively, you can use the **ceph mds stat** command to check if the MDS is in an active state:

  Syntax

```
ceph mds stat
```

  Example

```
[ansible@admin ceph-ansible]$ ceph mds stat
cephfs:1 {0=node01=up:active} 1 up:standby
```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- See the *Removing a Ceph MDS using Ansible* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more details on removing an MDS using Ansible.

1.4.2. Adding a Ceph MDS using the command-line interface

You can manually add a Ceph Metadata Server (MDS) using the command-line interface.

Prerequisites

- The **ceph-common** package is installed.
- A running Red Hat Ceph Storage cluster.
- **Root** or **sudo** access to the MDS nodes.
- New or existing servers that can be provisioned as MDS nodes.

Procedure

1. Add a new MDS node by logging into the node and creating an MDS mount point:

   Syntax

   ```
   sudo mkdir /var/lib/ceph/mds/ceph-MDS_ID
   ```

   Replace *MDS_ID* with the ID of the MDS node that you want to add the MDS daemon to.

   Example

   ```
   [admin@node03 ~]$ sudo mkdir /var/lib/ceph/mds/ceph-node03
   ```

2. If this is a new MDS node, create the authentication key if you are using Cephx authentication:

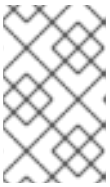
   Syntax

   ```
   sudo ceph auth get-or-create mds.MDS_ID mon 'profile mds' mgr 'profile mds' mds 'allow *' osd 'allow *' > /var/lib/ceph/mds/ceph-MDS_ID/keyring
   ```

   Replace *MDS_ID* with the ID of the MDS node to deploy the MDS daemon on.

   Example

   ```
   [admin@node03 ~]$ sudo ceph auth get-or-create mds.node03 mon 'profile mds' mgr 'profile mds' mds 'allow *' osd 'allow *' > /var/lib/ceph/mds/ceph-node03/keyring
   ```

   NOTE

   Cephx authentication is enabled by default. See the *Cephx authentication* link in the *Additional Resources* section for more information about Cephx authentication.

3. Start the MDS daemon:

   Syntax

   ```
   sudo systemctl start ceph-mds@HOST_NAME
   ```

   Replace *HOST_NAME* with the short name of the host to start the daemon.

   Example

```
[admin@node03 ~]$ sudo systemctl start ceph-mds@node03
```

4. Enable the MDS service:

   Syntax

   ```
   systemctl enable ceph-mds@HOST_NAME
   ```

   Replace *HOST_NAME* with the short name of the host to enable the service.

   Example

   ```
   [admin@node03 ~]$ sudo systemctl enable ceph-mds@node03
   ```

Verification

- Check the status of the MDS daemons:

  Syntax

  ```
  ceph fs dump
  ```

  Example

  ```
  [admin@mon]$ ceph fs dump

  [mds.node01 {0:115304} state up:active seq 5 addr
  [v2:172.25.250.10:6800/695510951,v1:172.25.250.10:6801/695510951]]

  Standby daemons:
  [mds.node03 {-1:144437} state up:standby seq 2 addr
  [v2:172.25.250.11:6800/172950087,v1:172.25.250.11:6801/172950087]]
  ```

- Alternatively, you can use the **ceph mds stat** command to check if the MDS is in an active state:

  Syntax

  ```
  ceph mds stat
  ```

  Example

  ```
  [ansible@admin ceph-ansible]$ ceph mds stat
  cephfs:1 {0=node01=up:active} 1 up:standby
  ```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- For more information on Cephx authentication, see the *Red Hat Ceph Storage Configuration Guide*.

- See the *Removing a Ceph MDS using the command line interface* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more details on removing an MDS using the command line interface.

1.4.3. Removing a Ceph MDS using Ansible

To remove a Ceph Metadata Server (MDS) using Ansible, use the **shrink-mds** playbook.

NOTE

If there is no replacement MDS to take over once the MDS is removed, the file system will become unavailable to clients. If that is not desirable, consider adding an additional MDS before removing the MDS you would like to take offline.

Prerequisites

- At least one MDS node.
- A running Red Hat Ceph Storage cluster deployed by Ansible.
- **Root** or **sudo** access to an Ansible administration node.

Procedure

1. Log in to the Ansible administration node.
2. Change to the **/usr/share/ceph-ansible** directory:

   Example

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

3. Run the Ansible **shrink-mds.yml** playbook, and when prompted, type **yes** to confirm shrinking the cluster:

   Syntax

   ```
   ansible-playbook infrastructure-playbooks/shrink-mds.yml -e mds_to_kill=ID -i hosts
   ```

   Replace *ID* with the ID of the MDS node you want to remove. You can remove only one Ceph MDS each time the playbook runs.

   Example

   ```
   [ansible @admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/shrink-mds.yml -e mds_to_kill=node02 -i hosts
   ```

4. As **root** or with **sudo** access, open and edit the **/usr/share/ceph-ansible/hosts** inventory file and remove the MDS node under the **[mdss]** section:

   Syntax

```
[mdss]
MDS_NODE_NAME
MDS_NODE_NAME
```

Example

```
[mdss]
node01
node03
```

In this example, **node02** was removed from the **[mdss]** list.

Verification

- Check the status of the MDS daemons:

  Syntax

  ```
  ceph fs dump
  ```

  Example

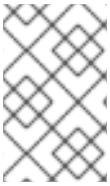
  ```
  [ansible@admin ceph-ansible]$ ceph fs dump

  [mds.node01 {0:115304} state up:active seq 5 addr
  [v2:172.25.250.10:6800/695510951,v1:172.25.250.10:6801/695510951]]

  Standby daemons:
  [mds.node03 {-1:144437} state up:standby seq 2 addr
  [v2:172.25.250.11:6800/172950087,v1:172.25.250.11:6801/172950087]]
  ```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- See the *Adding a Ceph MDS using Ansible* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more details on adding an MDS using Ansible.

1.4.4. Removing a Ceph MDS using the command-line interface

You can manually remove a Ceph Metadata Server (MDS) using the command-line interface.



NOTE

If there is no replacement MDS to take over once the current MDS is removed, the file system will become unavailable to clients. If that is not desirable, consider adding an MDS before removing the existing MDS.

Prerequisites

- The **ceph-common** package is installed.

- A running Red Hat Ceph Storage cluster.
- **Root** or **sudo** access to the MDS nodes.

Procedure

1. Log into the Ceph MDS node that you want to remove the MDS daemon from.
2. Stop the Ceph MDS service:

   Syntax

   ```
   sudo systemctl stop ceph-mds@HOST_NAME
   ```

   Replace *HOST_NAME* with the short name of the host where the daemon is running.

   Example

   ```
   [admin@node02 ~]$ sudo systemctl stop ceph-mds@node02
   ```

3. Disable the MDS service if you are not redeploying MDS to this node:

   Syntax

   ```
   sudo systemctl disable ceph-mds@HOST_NAME
   ```

   Replace *HOST_NAME* with the short name of the host to disable the daemon.

   Example

   ```
   [admin@node02 ~]$ sudo systemctl disable ceph-mds@node02
   ```

4. Remove the **/var/lib/ceph/mds/ceph-*MDS_ID*** directory on the MDS node:

   Syntax

   ```
   sudo rm -fr /var/lib/ceph/mds/ceph-MDS_ID
   ```

   Replace *MDS_ID* with the ID of the MDS node that you want to remove the MDS daemon from.

   Example

   ```
   [admin@node02 ~]$ sudo rm -fr /var/lib/ceph/mds/ceph-node02
   ```

Verification

- Check the status of the MDS daemons:

  Syntax

  ```
  ceph fs dump
  ```

Example

```
[ansible@admin ceph-ansible]$ ceph fs dump

[mds.node01 {0:115304} state up:active seq 5 addr
[v2:172.25.250.10:6800/695510951,v1:172.25.250.10:6801/695510951]]

Standby daemons:
[mds.node03 {-1:144437} state up:standby seq 2 addr
[v2:172.25.250.11:6800/172950087,v1:172.25.250.11:6801/172950087]]
```

Additional Resources

- For more information on installing Red Hat Ceph Storage, see the *Red Hat Ceph Storage Installation Guide*.
- See the *Adding a Ceph MDS using the command line interface* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more details on adding an MDS using the command line interface.

1.5. 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.

1.5.1. Ceph OSD node configuration

Configure Ceph OSDs and their supporting hardware similarly 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.

If you add drives of dissimilar size, 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*.

Additional Resources

- See the *Red Hat Ceph Storage Storage Strategies Guide* for more details. *

1.5.2. 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 reference the drive it uses.

IMPORTANT

The examples below are running on Red Hat Enterprise Linux 8. In Red Hat Enterprise Linux 8, **podman** is the default service and has replaced the older **docker** service. If you are running on Red Hat Enterprise Linux 7, then substitute **podman** with **docker** to execute the commands given.

Prerequisites

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

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 **podman ps** to list all containers:

   Example

```
[root@ceph3 ~]# podman ps
CONTAINER ID  IMAGE                                                 COMMAND           CREATED
STATUS            PORTS             NAMES
3a866f927b74  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  About
an hour ago  Up About an hour                ceph-osd-ceph3-sdd
4e242d932c32  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  About
an hour ago  Up About an hour                ceph-osd-ceph3-sdc
91f3d4829079  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  22
hours ago    Up 22 hours                    ceph-osd-ceph3-sdb
73dfe4021a49  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
ago          Up 7 days                      ceph-osd-ceph3-sdf
90f6d756af39  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
ago          Up 7 days                      ceph-osd-ceph3-sde
e66d6e33b306  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
ago          Up 7 days                      ceph-mgr-ceph3
733f37aafd23  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
ago          Up 7 days                      ceph-mon-ceph3
```

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

Example

```
[root@ceph3 ~]# podman 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.

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

For Ceph OSDs with the same disk topology, Ansible adds 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.yml** file.

NOTE

The new Ceph OSD nodes 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 *Red Hat Ceph Storage Installation Guide*.
- 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:

   Syntax

   ```
   [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. For **bare-metal** and **containers** deployments, run the **add-osd.yml** Ansible playbook:

   NOTE

   For a **new** OSD host, you need to run either the **site.yml** or the **site-container.yml** playbook with the **--limit** option as **node-exporter** and **ceph-crash** services are not deployed on the node with **osds.yml** playbook.

   Example

   ```
   [user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i hosts
   ```

   For **new** OSD host, run the **site.yml** or **site-container.yml** Ansible playbook:

   - **Bare-metal** deployments:

     Syntax

     ```
     ansible-playbook site.yml -i hosts --limit NEW_OSD_NODE_NAME
     ```

     Example

     ```
     [user@admin ceph-ansible]$ ansible-playbook site.yml -i hosts --limit node03
     ```

   - **Container** deployments:

Syntax

```
ansible-playbook site-container.yml -i hosts --limit NEW_OSD_NODE_NAME
```

Example

```
[user@admin ceph-ansible]$ ansible-playbook site-container.yml -i hosts --limit node03
```

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
```

Additional Resources

- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

1.5.4. 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 *Red Hat Ceph Storage Installation Guide*.
- 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
```

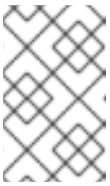
d. Verify that Ansible can reach all the Ceph nodes:

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

e. Change directory to the Ansible configuration directory:

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

f. For **bare-metal** and **containers** deployments, run the **add-osd.yml** Ansible playbook:



NOTE

For a **new** OSD host, you need to run either the **site.yml** or the **site-container.yml** playbook with the **--limit** option as **node-exporter** and **ceph-crash** services are not deployed on the node with **osds.yml** playbook.

Example

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i
hosts
```

For **new** OSD host, run the **site.yml** or **site-container.yml** Ansible playbook:

- **Bare-metal** deployments:

  Syntax

```
ansible-playbook site.yml -i hosts --limit NEW_OSD_NODE_NAME
```

Example

```
[user@admin ceph-ansible]$ ansible-playbook site.yml -i hosts --limit node03
```

- **Container** deployments:

  Syntax

  ```
  ansible-playbook site-container.yml -i hosts --limit NEW_OSD_NODE_NAME
  ```

  Example

  ```
  [user@admin ceph-ansible]$ ansible-playbook site-container.yml -i hosts --limit
  node03
  ```

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']"
   ```

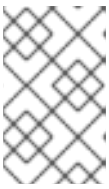
   b. Verify that Ansible can reach the all Ceph nodes:

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

   c. Change directory to the Ansible configuration directory:

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

3. For **bare-metal** and **containers** deployments, run the **add-osd.yml** Ansible playbook:



NOTE

For a **new** OSD host, you need to run either the **site.yml** or the **site-container.yml** playbook with the **--limit** option as **node-exporter** and **ceph-crash** services are not deployed on the node with **osds.yml** playbook.

Example

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i hosts
```

For **new** OSD host, run the **site.yml** or **site-container.yml** Ansible playbook:

- **Bare-metal** deployments:

  Syntax

  ```
  ansible-playbook site.yml -i hosts --limit NEW_OSD_NODE_NAME
  ```

  Example

  ```
  [user@admin ceph-ansible]$ ansible-playbook site.yml -i hosts --limit node03
  ```

- **Container** deployments:

  Syntax

  ```
  ansible-playbook site-container.yml -i hosts --limit NEW_OSD_NODE_NAME
  ```

  Example

  ```
  [user@admin ceph-ansible]$ ansible-playbook site-container.yml -i hosts --limit node03
  ```

Additional Resources

- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

1.5.5. Creating Ceph OSDs using ceph-volume

The **create** subcommand calls the **prepare** subcommand, and then calls the **activate** subcommand.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph OSD nodes.

NOTE

If you prefer to have more control over the creation process, you can use the **prepare** and **activate** subcommands separately to create the OSD, instead of using **create**. You can use the two subcommands to gradually introduce new OSDs into a storage cluster, while avoiding having to rebalance large amounts of data. Both approaches work the same way, except that using the **create** subcommand causes the OSD to become *up* and *in* immediately after completion.

Procedure

1. To create a new OSD:

   Syntax

   ```
   ceph-volume lvm create --bluestore --data VOLUME_GROUP/LOGICAL_VOLUME
   ```

Example

```
[root@osd ~]# ceph-volume lvm create --bluestore --data example_vg/data_lv
```

Additional Resources

- See the *Preparing Ceph OSDs using `ceph-volume`* section in the *Red Hat Ceph Storage Administration Guide* for more details.
- See the *Activating Ceph OSDs using `ceph-volume`* section in the *Red Hat Ceph Storage Administration Guide* for more details.

1.5.6. Using batch mode with ceph-volume

The **batch** subcommand automates the creation of multiple OSDs when single devices are provided.

The **ceph-volume** command decides the best method to use to create the OSDs, based on drive type. Ceph OSD optimization depends on the available devices:

- If all devices are traditional hard drives, **batch** creates one OSD per device.
- If all devices are solid state drives, **batch** creates two OSDs per device.
- If there is a mix of traditional hard drives and solid state drives, **batch** uses the traditional hard drives for data, and creates the largest possible journal (**block.db**) on the solid state drive.

NOTE

The **batch** subcommand does not support the creation of a separate logical volume for the write-ahead-log (**block.wal**) device.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph OSD nodes.

Procedure

1. To create OSDs on several drives:

Syntax

```
ceph-volume lvm batch --bluestore PATH_TO_DEVICE [PATH_TO_DEVICE]
```

Example

```
[root@osd ~]# ceph-volume lvm batch --bluestore /dev/sda /dev/sdb /dev/nvme0n1
```

Additional Resources

- See the *Creating Ceph OSDs using `ceph-volume`* section in the *Red Hat Ceph Storage Administration Guide* for more details.

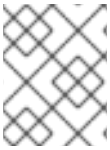
1.5.7. Adding a Ceph OSD using the command-line interface

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. Create volume groups and logical volumes.
4. Add the new OSD node to the CRUSH map.
5. Update the owner and group permissions.
6. 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.



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 *Red Hat Ceph Storage Installation Guide*.
- The **root** access to the new nodes.
- Optional. If you do not want the **ceph-volume** utility to create a volume group and logical volumes automatically, create them manually. See the *Configuring and managing logical volumes* guide for Red Hat Enterprise Linux 8.

Procedure

1. Enable the Red Hat Ceph Storage 4 OSD software repository.

   Red Hat Enterprise Linux 7

   ```
   [root@osd ~]# subscription-manager repos --enable=rhel-7-server-rhceph-4-osd-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   [root@osd ~]# subscription-manager repos --enable=rhceph-4-osd-for-rhel-8-x86_64-rpms
   ```

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

```
[root@osd ~]# 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.keyring /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 7

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

Red Hat Enterprise Linux 8

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

5. Prepare the OSDs.

- To use previously created logical volumes:

  Syntax

  ```
  ceph-volume lvm prepare --bluestore --data VOLUME_GROUP/LOGICAL_VOLUME
  ```

- To specify a raw device for **ceph-volume** to create logical volumes automatically:

  Syntax

  ```
  ceph-volume lvm prepare --bluestore --data /PATH_TO_DEVICE
  ```

  See the *Preparing OSDs* section for more details.

6. Set the **noup** option:

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

7. Activate the new OSD:

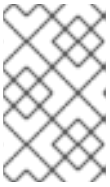
Syntax

```
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

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

See the *Activating OSDs* section for more details.



NOTE

You can prepare and activate OSDs with a single command. See the *Creating OSDs* section for details. Alternatively, you can specify multiple drives and create OSDs with a single command. See the *Using **batch** mode*.

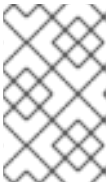
8. Add the OSD to the CRUSH map. If you specify more than one bucket, the command places the OSD into the most specific bucket out of those you specified, *and* it moves the bucket underneath any other buckets you specified.

Syntax

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

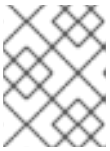
Example

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



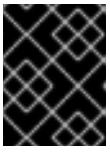
NOTE

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 *Red Hat Ceph Storage Storage Strategies Guide*.



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 storage clusters with custom names, then add the following line to the appropriate file:

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

Replace ***CLUSTER_NAME*** with the custom storage 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

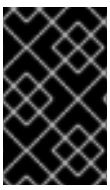
```
[root@osd ~]# systemctl enable ceph-osd@4
[root@osd ~]# systemctl start ceph-osd@4
```

Additional Resources

- See the *Editing a CRUSH map* section in the *Red Hat Ceph Storage Storage Strategies Guide* for more information.
- See the *Red Hat Ceph Storage Administration Guide*, for more information on using the **ceph-volume** command.

1.5.8. Adding a Ceph OSD using the command-line interface in a containerized environment

You can manually add a single or multiple Ceph OSD using the command-line interface in a containerized Red Hat Ceph Storage cluster.



IMPORTANT

Red Hat recommends the use of **ceph-ansible** to add Ceph OSDs unless there is an exception or a specific use case where adding Ceph OSDs manually is required. If you are not sure, contact Red Hat Support.

Prerequisites

- A running Red Hat Ceph Storage cluster in a containerized environment.

- Having root access to the container node.
- An existing OSD node.

IMPORTANT

The examples below are running on Red Hat Enterprise Linux 8. In Red Hat Enterprise Linux 8, podman is the default service and has replaced the older docker service. If you are running on Red Hat Enterprise Linux 7, then substitute podman with docker to execute the commands given.

Procedure

1. To create a single OSD, execute the **lvm prepare** command:

   Syntax

   ```
   podman run --rm --net=host --privileged=true --pid=host --ipc=host -v /dev:/dev -v
   /etc/localtime:/etc/localtime:ro -v /var/lib/ceph:/var/lib/ceph:z -v /etc/ceph:/etc/ceph:z -v
   /var/run/ceph:/var/run/ceph:z -v /var/run/udev/:/var/run/udev/ -v /var/log/ceph:/var/log/ceph:z -
   v /run/lvm/:/run/lvm/ --entrypoint=ceph-volume PATH_TO_IMAGE --cluster CLUSTER_NAME
   lvm prepare --bluestore --data PATH_TO_DEVICE --no-systemd
   ```

   Example

   ```
   [root@osd ~]# podman run --rm --net=host --privileged=true --pid=host --ipc=host -v
   /dev:/dev -v /etc/localtime:/etc/localtime:ro -v /var/lib/ceph:/var/lib/ceph:z -v
   /etc/ceph:/etc/ceph:z -v /var/run/ceph:/var/run/ceph:z -v /var/run/udev/:/var/run/udev/ -v
   /var/log/ceph:/var/log/ceph:z -v /run/lvm/:/run/lvm/ --entrypoint=ceph-volume
   registry.redhat.io/rhceph/rhceph-4-rhel8:latest --cluster ceph lvm prepare --bluestore --data
   /dev/sdh --no-systemd
   ```

   The example prepares a single Bluestore Ceph OSD with data on **/dev/sdh**.

   NOTE

   To enable and start the OSD, execute the following commands:

   Example

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

   You can also use the following optional arguments:

   dmcrypt

   Description

   Enable encryption for the underlying OSD devices.

   block.db

   Description

Path to a bluestore block.db logical volume or partition.

block.wal

Description

Path to a bluestore block.wal logical volume or partition.

2. To create multiple Ceph OSDs, execute the **lvm batch** command:

Syntax

```
podman run --rm --net=host --privileged=true --pid=host --ipc=host -v /dev:/dev -v /etc/localtime:/etc/localtime:ro -v /var/lib/ceph:/var/lib/ceph:z -v /etc/ceph:/etc/ceph:z -v /var/run/ceph:/var/run/ceph:z -v /var/run/udev/:/var/run/udev/ -v /var/log/ceph:/var/log/ceph:z -v /run/lvm/:/run/lvm/ --entrypoint=ceph-volume PATH_TO_IMAGE --cluster CLUSTER_NAME lvm batch --bluestore --yes --prepare _PATH_TO_DEVICE PATH_TO_DEVICE --no-systemd
```

Example

```
[root@osd ~]# podman run --rm --net=host --privileged=true --pid=host --ipc=host -v /dev:/dev -v /etc/localtime:/etc/localtime:ro -v /var/lib/ceph:/var/lib/ceph:z -v /etc/ceph:/etc/ceph:z -v /var/run/ceph:/var/run/ceph:z -v /var/run/udev/:/var/run/udev/ -v /var/log/ceph:/var/log/ceph:z -v /run/lvm/:/run/lvm/ --entrypoint=ceph-volume registry.redhat.io/rhceph/rhceph-4-rhel8:latest --cluster ceph lvm batch --bluestore --yes --prepare /dev/sde /dev/sdf --no-systemd
```

The example prepares multiple Bluestore Ceph OSDs with data on **/dev/sde** and **/dev/sdf** .

You can also use the following optional arguments:

dmcrypt

Description

Enable encryption for the underlying OSD devices.

db-devices

Description

Path to a bluestore block.db logical volume or partition.

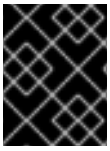
wal-devices

Description

Path to a bluestore block.wal logical volume or partition.

1.5.9. 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, run the **shrink-osd.yml** playbook.



IMPORTANT

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

IMPORTANT

Before removing OSDs, verify that the cluster has enough space to re-balance.

IMPORTANT

Do not remove OSDs simultaneously unless you are sure the placement groups are in an **active+clean** state and the OSDs do not contain replicas or erasure coding shards for the same objects. If unsure, contact Red Hat Support.

Prerequisites

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

Procedure

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

   Syntax

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

2. Copy the admin keyring from **/etc/ceph/** on the Ceph Monitor node to the node that contains the OSD that you want to remove.

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

   Syntax

   ```
   ansible-playbook infrastructure-playbooks/shrink-osd.yml -e osd_to_kill=ID -u
   ANSIBLE_USER_NAME -i hosts
   ```

   Replace:

   - ***ID*** with the ID of the OSD node. To remove multiple OSDs, separate the OSD IDs with a comma.
   - ***ANSIBLE_USER_NAME*** with the name of the Ansible user.

   Example

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

4. Verify that the OSD has been successfully removed:

   Syntax

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

Additional Resources

- The *Red Hat Ceph Storage Installation Guide*.
- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

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

Removing an OSD from a storage cluster involves these steps: * Updating the cluster map. * Removing its authentication key. * Removing the OSD from the OSD map. * Removing the OSD from the **ceph.conf** file.

If the OSD node has multiple drives, you might need to remove an OSD for each drive by repeating this procedure for each OSD that you want to remove.

Prerequisites

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

Procedure

1. 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**.

2. Remove the OSD from the storage cluster:

   Syntax

   ```
   ceph osd out OSD_ID
   ```

   Example

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

IMPORTANT

Once the OSD has been removed, Ceph starts rebalancing and copying data to the remaining 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:

Syntax

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

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

   To manually remove the OSD and the bucket that contains it, 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 instructions for decompilimg a CRUSH map in 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. The default name for the file is **/etc/ceph/ceph.conf**. Remove the OSD entry in the file, 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/
   ```

1.5.11. Replacing a BlueStore database disk using the command-line interface

When replacing the BlueStore DB device, **block.db**, that contains the BlueStore OSD's internal metadata, Red Hat supports the re-deploying of all OSDs using Ansible and the command-line interface (CLI). A corrupt **block.db** file will impact all OSDs which are included in that **block.db** files.

The procedure to replace the BlueStore **block.db** disk, is to mark out each device in turn, wait for the data to replicate across the cluster, replace the OSD, and mark it back in again. You can retain the ***OSD_ID*** and recreate OSD with the new **block.db** partition on the replaced disk. Although this is a simple procedure. it requires a lot of data migration.

NOTE

If the **block.db** device has multiple OSDs, then follow this procedure for each of the OSDs on the **block.db** device. You can run **ceph-volume lvm list** to see **block.db** to block relationships.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A storage device with partition.
- Root-level access to all the nodes.

Procedure

1. Check current Ceph cluster status on the monitor node:

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

2. Identify the failed OSDs to replace:

```
[root@mon ~]# ceph osd tree | grep -i down
```

3. Stop and disable OSD service on OSD node:

Syntax

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

Example

```
[root@osd1 ~]# systemctl stop ceph-osd@1
[root@osd1 ~]# systemctl disable ceph-osd@1
```

4. Set OSD **out** on the monitor node:

Syntax

```
ceph osd out OSD_ID
```

Example

```
[root@mon ~]# ceph osd out 1
```

5. Wait for the data to migrate off the OSD:

Syntax

```
while ! ceph osd safe-to-destroy OSD_ID ; do sleep 60 ; done
```

Example

```
[root@mon ~]# while ! ceph osd safe-to-destroy 1 ; do sleep 60 ; done
```

6. Stop the OSD daemon on the OSD node:

Syntax

```
systemctl kill ceph-osd@OSD_ID
```

Example

```
[root@osd1 ~]# systemctl kill ceph-osd@1
```

7. Make note of which device this OSD is using:

Syntax

```
mount | grep /var/lib/ceph/osd/ceph-OSD_ID
```

-

Example

```
[root@osd1 ~]# mount | grep /var/lib/ceph/osd/ceph-1
```

8. Unmount mount point of the failed drive path on OSD node:

Syntax

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

Example

```
[root@osd1 ~] #umount /var/lib/ceph/osd/ceph-1
```

9. Set the **noout** and **norebalance** to avoid backfilling and re-balancing:

```
[root@mon ~]# ceph osd set noout
[root@mon ~]# ceph osd set norebalance
```

10. Replace the physical drive. Refer to the hardware vendor's documentation for the node. Allow the new drive to appear under the /**dev**/ directory and make a note of the drive path before proceeding further.

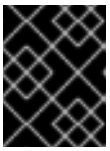
11. Destroy OSDs on the monitor node:

Syntax

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

Example

```
[root@mon ~]# ceph osd destroy 1 --yes-i-really-mean-it
```



IMPORTANT

This step destroys the contents of the device. Ensure the data on the device is not needed and the cluster is healthy.

12. Remove the logical volume manager on the OSD disk:

Syntax

```
lvremove /dev/VOLUME_GROUP/LOGICAL_VOLUME
vgremove VOLUME_GROUP
pvremove /dev/DEVICE
```

Example

```
[root@osd1 ~]# lvremove /dev/data-vg1/data-lv1
[root@osd1 ~]# vgremove data-vg1
[root@osd1 ~]# pvremove /dev/sdb
```

13. Zap the OSD disk on OSD node:

Syntax

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

Example

```
[root@osd1 ~]# ceph-volume lvm zap /dev/sdb
```

14. Recreate lvm on OSD disk:

Syntax

```
pvcreate /dev/DEVICE
vgcreate VOLUME_GROUP /dev/DEVICE
lvcreate -l SIZE -n LOGICAL_VOLUME VOLUME_GROUP
```

Example

```
[root@osd1 ~]# pvcreate /dev/sdb
[root@osd1 ~]# vgcreate data-vg1 /dev/sdb
[root@osd1 ~]# lvcreate -l 100%FREE -n data-lv1 data-vg1
```

15. Create lvm on the new **block.db** disk:

Syntax

```
pvcreate /dev/DEVICE
vgcreate VOLUME_GROUP_DATABASE /dev/DEVICE
lvcreate -LI SIZE -n LOGICAL_VOLUME_DATABASE VOLUME_GROUP_DATABASE
```

Example

```
[root@osd1 ~]# pvcreate /dev/sdb
[root@osd1 ~]# vgcreate db-vg1 /dev/sdb
[root@osd1 ~]# lvcreate -l 100%FREE -n lv-db1 db-vg1
```

16. Recreate the OSDs on the OSD node:

Syntax

```
ceph-volume lvm create --bluestore --osd-id OSD_ID --data
VOLUME_GROUP/LOGICAL_VOLUME --block.db
VOLUME_GROUP_DATABASE/LOGICAL_VOLUME_DATABASE
```

Example

```
[root@osd1 ~]# ceph-volume lvm create --bluestore --osd-id 1 --data data-vg1/data-lv1 --
block.db db-vg1/db-lv1
```

NOTE

Red Hat recommends to use the same *OSD_ID* as the one destroyed in the previous steps.

17. Start and enable OSD service on OSD node:

Syntax

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

Example

```
[root@osd1 ~]# systemctl start ceph-osd@1
[root@osd1 ~]# systemctl enable ceph-osd@1
```

18. Check the CRUSH hierarchy to ensure OSD is in the cluster:

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

19. Unset noout and norebalance:

```
[root@mon ~]# ceph osd unset noout
[root@mon ~]# ceph osd unset norebalance
```

20. Monitor cluster status until **HEALTH_OK**:

```
[root@mon ~]# watch -n2 ceph -s
```

Additional Resources

- See the *Installing a Red Hat Ceph Storage cluster* chapter in *Red Hat Ceph StorageInstallation Guide* for more information.

1.5.12. 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:

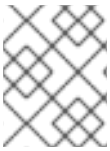
```
[root@monitor ~]# ceph -w
```

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**.

1.6. RECALCULATING THE PLACEMENT GROUPS

Placement groups (PGs) define the spread of any pool data across the available OSDs. A placement group is built 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.



NOTE

See the KnowledgeBase article *How do I increase placement group (PG) count in a Ceph Cluster* for additional details.

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.

1.7. USING THE CEPH MANAGER BALANCER MODULE

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

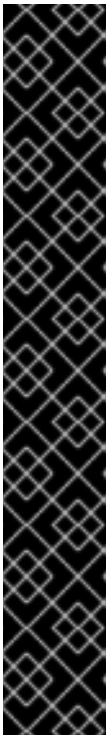
Modes

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

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

```
[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
```

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. Make sure that the balancer module is on:

   Example

   ```
   [root@mon ~]# ceph mgr module ls | more
   {
      "always_on_modules": [
         "balancer",
         "crash",
         "devicehealth",
         "orchestrator_cli",
         "progress",
         "rbd_support",
         "status",
         "volumes"
      ],
      "enabled_modules": [
         "dashboard",
         "pg_autoscaler",
         "prometheus"
      ],
   ```

   a. If the balancer module is not listed in the **always_on** or **enabled** modules, enable it:

      Syntax

      ```
      ceph mgr module enable balancer
      ```

2. Turn on the balancer module:

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

3. 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 **target_max_misplaced_ratio** setting. For example, to increase the threshold to 7%:

Example

```
[root@mon ~]# ceph config set mgr target_max_misplaced_ratio .07
```

For automatic balancing:

- Set the number of seconds to sleep in between runs of the automatic balancer:

Example

```
[root@mon ~]# ceph config set mgr mgr/balancer/sleep_interval 60
```

- Set the time of day to begin automatic balancing in HHMM format:

Example

```
[root@mon ~]# ceph config set mgr mgr/balancer/begin_time 0000
```

- Set the time of day to finish automatic balancing in HHMM format:

Example

```
[root@mon ~]# ceph config set mgr mgr/balancer/end_time 2359
```

- Restrict automatic balancing to this day of the week or later. Uses the same conventions as crontab, **0** is Sunday, **1** is Monday, and so on:

Example

```
[root@mon ~]# ceph config set mgr mgr/balancer/begin_weekday 0
```

- Restrict automatic balancing to this day of the week or earlier. This uses the same conventions as crontab, **0** is Sunday, **1** is Monday, and so on:

Example

```
[root@mon ~]# ceph config set mgr mgr/balancer/end_weekday 6
```

- Define the pool IDs to which the automatic balancing is limited. The default for this is an empty string, meaning all pools are balanced. The numeric pool IDs can be gotten with the **ceph osd pool ls detail** command:

Example

```
[root@mon ~]#  ceph config set mgr mgr/balancer/pool_ids 1,2,3
```

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:

     #### Syntax

     ```
     ceph balancer eval POOL_NAME
     ```

     #### Example

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

   - To see greater detail for the evaluation:

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

   - To generate a plan using the currently configured mode:

     #### Syntax

     ```
     ceph balancer optimize PLAN_NAME
     ```

     Replace *PLAN_NAME* with a custom plan name.

     #### Example

     ```
     [root@mon ~]# ceph balancer optimize rbd_123
     ```

   - To see the contents of a plan:

     #### Syntax

```
ceph balancer show PLAN_NAME
```

Example

```
[root@mon ~]# ceph balancer show rbd_123
```

- To discard old plans:

  Syntax

  ```
  ceph balancer rm PLAN_NAME
  ```

  Example

  ```
  [root@mon ~]# ceph balancer rm rbd_123
  ```

- 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:

  Syntax

  ```
  ceph balancer eval PLAN_NAME
  ```

  Example

  ```
  [root@mon ~]# ceph balancer eval rbd_123
  ```

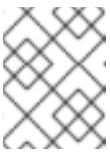
- To execute the plan:

  Syntax

  ```
  ceph balancer execute PLAN_NAME
  ```

  Example

  ```
  [root@mon ~]# ceph balancer execute rbd_123
  ```



NOTE

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

1.8. USING UPMAP TO MANUALLY REBALANCE DATA ON OSDS

As a storage administrator, you can manually rebalance data on OSDs by moving selected placement groups (PGs) to specific OSDs. To perform manual rebalancing, turn off the Ceph Manager balancer module and use the **upmap** mode to move the PGs.

Prerequisites

- A running Red Hat storage cluster.
- Root-level access to all nodes in the storage cluster.

Procedure

1. Make sure that the balancer module is on:

   Example

   ```
   [root@mon ~]# ceph mgr module ls | more
   {
       "always_on_modules": [
           "balancer",
           "crash",
           "devicehealth",
           "orchestrator_cli",
           "progress",
           "rbd_support",
           "status",
           "volumes"
       ],
       "enabled_modules": [
           "dashboard",
           "pg_autoscaler",
           "prometheus"
       ],
   ```

   a. If the balancer module is not listed in the **always_on** or **enabled** modules, enable it:

      Syntax

      ```
      ceph mgr module enable balancer
      ```

2. Set the balancer mode to **upmap**:

   Syntax

   ```
   ceph balancer mode upmap
   ```

3. Turn off the balancer module:

   Syntax

   ```
   ceph balancer off
   ```

4. Check balancer status:

   Example

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

```
    "plans": [],
    "active": false,
    "last_optimize_started": "",
    "last_optimize_duration": "",
    "optimize_result": "",
    "mode": "upmap"
}
```

5. Set the **norebalance** flag for the OSDs:

   Syntax

   ```
   ceph osd set norebalance
   ```

6. Use the **ceph pg dump pgs_brief** command to list the pools in your storage cluster and the space each consumes. Use **grep** to search for remapped pools.

   Example

   ```
   [root@mon ~]# ceph pg dump pgs_brief
   PG_STAT STATE                      UP            UP_PRIMARY ACTING
   ACTING_PRIMARY
   dumped pgs_brief
   7.270   active+remapped+backfilling  [8,48,61]         8 [46,48,61]           46
   7.1e7   active+remapped+backfilling [73,64,74]        73 [18,64,74]           18
   7.1c1   active+remapped+backfilling  [29,14,8]        29 [29,14,24]           29
   7.17f   active+remapped+backfilling [73,71,50]        73 [50,71,69]           50
   7.16c   active+remapped+backfilling   [66,8,4]        66  [66,4,57]           66
   7.13d   active+remapped+backfilling [73,27,56]        73 [27,56,35]           27
   7.130   active+remapped+backfilling [53,47,73]        53 [53,47,72]           53
   9.e0    active+remapped+backfilling  [8,75,14]        8 [14,75,58]           14
   7.db    active+remapped+backfilling [10,57,60]        10 [10,60,50]           10
   9.7     active+remapped+backfilling [26,69,38]        26 [26,38,41]           26
   7.4a    active+remapped+backfilling [73,10,76]        73 [10,76,29]           10
   9.9a    active+remapped+backfilling [20,15,73]        20 [20,15,29]           20
   7.ac    active+remapped+backfilling   [8,74,3]        8  [3,74,37]            3
   9.c2    active+remapped+backfilling  [57,75,7]        57   [4,75,7]            4
   7.34d   active+remapped+backfilling [23,46,73]        23 [23,46,56]           23
   7.36a   active+remapped+backfilling  [40,32,8]        40 [40,32,44]           40
   ```

7. Move the PGs to the OSDs where you want them to reside. For example, to move PG 7.ac from OSDs 8 and 3 to OSDs 3 and 37:

   Example

   ```
   PG_STAT STATE                      UP            UP_PRIMARY ACTING
   ACTING_PRIMARY
   dumped pgs_brief
   7.ac    active+remapped+backfilling   [8,74,3]         8  [3,74,37]            3

   [root@mon ~]# ceph osd pg-upmap-items 7.ac 8 3 3 37

   7.ac   active+clean               [3,74,37]         8 [3,74,37]            3
   ```

NOTE

Repeat this step to move each of the remapped PGs, one at a time.

8. Use the **ceph pg dump pgs_brief** command again to check that the PGs move to the **active+clean** state:

   Example

```
[root@mon ~]# ceph pg dump pgs_brief
PG_STAT STATE                 UP          UP_PRIMARY ACTING
ACTING_PRIMARY
dumped pgs_brief
7.270   active+clean          [8,48,61]       8 [46,48,61]          46
7.1e7   active+clean          [73,64,74]      73 [18,64,74]         18
7.1c1   active+clean          [29,14,8]       29 [29,14,24]         29
7.17f   active+clean          [73,71,50]      73 [50,71,69]         50
7.16c   active+clean          [66,8,4]        66  [66,4,57]         66
7.13d   active+clean          [73,27,56]      73 [27,56,35]         27
7.130   active+clean          [53,47,73]      53 [53,47,72]         53
9.e0    active+clean          [8,75,14]       8 [14,75,58]          14
7.db    active+clean          [10,57,60]      10 [10,60,50]         10
9.7     active+clean          [26,69,38]      26 [26,38,41]         26
7.4a    active+clean          [73,10,76]      73 [10,76,29]         10
9.9a    active+clean          [20,15,73]      20 [20,15,29]         20
7.ac    active+clean          [3,74,37]       8 [3,74,37]           3
9.c2    active+clean          [57,75,7]       57 [4,75,7]           4
7.34d   active+clean          [23,46,73]      23 [23,46,56]         23
7.36a   active+clean          [40,32,8]       40 [40,32,44]         40
```

   The time it takes for the PGs to move to **active+clean** depends on the numbers of PGs and OSDs. In addition, the number of objects misplaced depends on the value set for **mgr target_max_misplaced_ratio**. A higher value set for **target_max_misplaced_ratio** results in a greater number of misplaced objects; thus, it takes a longer time for all PGs to become **active+clean**.

9. Unset the **norebalance** flag:

   Syntax

```
ceph osd unset norebalance
```

10. Turn the balancer module back on:

    Syntax

```
ceph balancer on
```

Once you enable the balancer module, it slowly moves the PGs back to their intended OSDs according to the CRUSH rules for the storage cluster. The balancing process might take some time, but completes eventually.

1.9. USING THE CEPH MANAGER ALERTS MODULE

You can use the Ceph Manager alerts module to send simple alert messages about the Red Hat Ceph Storage cluster's health by email.

NOTE

This module is not intended to be a robust monitoring solution. The fact that it is run as part of the Ceph cluster itself is fundamentally limiting in that a failure of the **ceph-mgr** daemon prevents alerts from being sent. This module can, however, be useful for standalone clusters that exist in environments where existing monitoring infrastructure does not exist.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor node.

Procedure

1. Enable the alerts module:

   Example

   ```
   [root@host01 ~]# ceph mgr module enable alerts
   ```

2. Ensure the alerts module is enabled:

   Example

   ```
   [root@host01 ~]# ceph mgr module ls | more
   {
       "always_on_modules": [
           "balancer",
           "crash",
           "devicehealth",
           "orchestrator_cli",
           "progress",
           "rbd_support",
           "status",
           "volumes"
       ],
       "enabled_modules": [
           "alerts",
           "dashboard",
           "pg_autoscaler",
           "nfs",
           "prometheus",
           "restful"
       ]
   ```

3. Configure the Simple Mail Transfer Protocol (SMTP):

   Syntax

```
ceph config set mgr mgr/alerts/smtp_host SMTP_SERVER
ceph config set mgr mgr/alerts/smtp_destination RECEIVER_EMAIL_ADDRESS
ceph config set mgr mgr/alerts/smtp_sender SENDER_EMAIL_ADDRESS
```

Example

```
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_host smtp.example.com
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_destination example@example.com
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_sender example2@example.com
```

4. Optional: By default, the alerts module uses SSL and port 465. To change that, set the **smtp_ssl** to **false**:

Syntax

```
ceph config set mgr mgr/alerts/smtp_ssl false
ceph config set mgr mgr/alerts/smtp_port PORT_NUMBER
```

Example

```
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_ssl false
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_port 587
```

5. Authenticate to the SMTP server:

Syntax

```
ceph config set mgr mgr/alerts/smtp_user USERNAME
ceph config set mgr mgr/alerts/smtp_password PASSWORD
```

Example

```
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_user admin1234
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_password admin1234
```

6. Optional: By default, SMTP **From** name is **Ceph**. To change that, set the **smtp_from_name** parameter:

Syntax

```
ceph config set mgr mgr/alerts/smtp_from_name CLUSTER_NAME
```

Example

```
[root@host01 ~]# ceph config set mgr mgr/alerts/smtp_from_name 'Ceph Cluster Test'
```

7. Optional: By default, the alerts module checks the storage cluster's health every minute, and sends a message when there is a change in the cluster health status. To change the frequency, set the **interval** parameter:

Syntax

```
ceph config set mgr mgr/alerts/interval INTERVAL
```

Example

```
[root@host01 ~]# ceph config set mgr mgr/alerts/interval "5m"
```

In this example, the interval is set to 5 minutes.

8. Optional: Send an alert immediately:

Example

```
[root@host01 ~]# ceph alerts send
```

Additional Resources

- See the *Health messages of a Ceph cluster* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more information on Ceph health messages.

1.10. USING THE CEPH MANAGER CRASH MODULE

Using the Ceph manager crash module, you can collect information about daemon crashdumps and store it in the Red Hat Ceph Storage cluster for further analysis.

By default, daemon crashdumps are dumped in **/var/lib/ceph/crash**. You can configure crashdumps with the option **crash dir**. Crash directories are named by time, date, and a randomly-generated UUID, and contain a metadata file **meta** and a recent log file, with a **crash_id** that is the same.

You can use **ceph-crash.service** to submit these crash automatically and persist in the Ceph Monitors. The **ceph-crash.service** watches watches the crashdump directory and uploads them with **ceph crash post**.

The *RECENT_CRASH* heath message is one of the most common health messages in a Ceph cluster. This health message means that one or more Ceph daemons has crashed recently, and the crash has not yet been archived or acknowledged by the administrator. This might indicate a software bug, a hardware problem like a failing disk, or some other problem. The option **mgr/crash/warn_recent_interval** controls the time period of what recent means, which is two weeks by default. You can disable the warnings by running the following command:

Example

```
[root@mon ~]# ceph config set mgr/crash/warn_recent_interval 0
```

The option **mgr/crash/retain_interval** controls the period for which you want to retain the crash reports before they are automatically purged. The default for this option is one year.

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. Ensure the crash module is enabled:

Example

```
[root@mon ~]# ceph mgr module ls | more
{
    "always_on_modules": [
        "balancer",
        "crash",
        "devicehealth",
        "orchestrator_cli",
        "progress",
        "rbd_support",
        "status",
        "volumes"
    ],
    "enabled_modules": [
        "dashboard",
        "pg_autoscaler",
        "prometheus"
    ]
```

2. Save a crash dump: The metadata file is a JSON blob stored in the crash dir as **meta**. You can invoke the ceph command **-i -** option, which reads from stdin.

   Example

   ```
   [root@mon ~]# ceph crash post -i meta
   ```

3. List the timestamp or the UUID crash IDs for all the new and archived crash info:

   Example

   ```
   [root@mon ~]# ceph crash ls
   ```

4. List the timestamp or the UUID crash IDs for all the new crash information:

   Example

   ```
   [root@mon ~]# ceph crash ls-new
   ```

5. List the timestamp or the UUID crash IDs for all the new crash information:

   Example

   ```
   [root@mon ~]# ceph crash ls-new
   ```

6. List the summary of saved crash information grouped by age:

   Example

   ```
   [root@mon ~]# ceph crash stat
   8 crashes recorded
   8 older than 1 days old:
   2021-05-20T08:30:14.533316Z_4ea88673-8db6-4959-a8c6-0eea22d305c2
   ```

```
2021-05-20T08:30:14.590789Z_30a8bb92-2147-4e0f-a58b-a12c2c73d4f5
2021-05-20T08:34:42.278648Z_6a91a778-bce6-4ef3-a3fb-84c4276c8297
2021-05-20T08:34:42.801268Z_e5f25c74-c381-46b1-bee3-63d891f9fc2d
2021-05-20T08:34:42.803141Z_96adfc59-be3a-4a38-9981-e71ad3d55e47
2021-05-20T08:34:42.830416Z_e45ed474-550c-44b3-b9bb-283e3f4cc1fe
2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
2021-05-24T19:58:44.315282Z_1847afbc-f8a9-45da-94e8-5aef0738954e
```

7. View the details of the saved crash:

Syntax

```
ceph crash info CRASH_ID
```

Example

```
[root@mon ~]# ceph crash info 2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-
9a59af7b7a2d
{
    "assert_condition": "session_map.sessions.empty()",
    "assert_file": "/builddir/build/BUILD/ceph-16.1.0-486-g324d7073/src/mon/Monitor.cc",
    "assert_func": "virtual Monitor::~Monitor()",
    "assert_line": 287,
    "assert_msg": "/builddir/build/BUILD/ceph-16.1.0-486-g324d7073/src/mon/Monitor.cc: In
function 'virtual Monitor::~Monitor()' thread 7f67a1aeb700 time 2021-05-
24T19:58:42.545485+0000\n/builddir/build/BUILD/ceph-16.1.0-486-
g324d7073/src/mon/Monitor.cc: 287: FAILED
ceph_assert(session_map.sessions.empty())\n",
    "assert_thread_name": "ceph-mon",
    "backtrace": [
        "/lib64/libpthread.so.0(+0x12b30) [0x7f679678bb30]",
        "gsignal()",
        "abort()",
        "(ceph::__ceph_assert_fail(char const*, char const*, int, char const*)+0x1a9)
[0x7f6798c8d37b]",
        "/usr/lib64/ceph/libceph-common.so.2(+0x276544) [0x7f6798c8d544]",
        "(Monitor::~Monitor()+0xe30) [0x561152ed3c80]",
        "(Monitor::~Monitor()+0xd) [0x561152ed3cdd]",
        "main()",
        "__libc_start_main()",
        "_start()"
    ],
    "ceph_version": "14.1.0-486.el8cp",
    "crash_id": "2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d",
    "entity_name": "mon.ceph-adm4",
    "os_id": "rhel",
    "os_name": "Red Hat Enterprise Linux",
    "os_version": "8.3 (Ootpa)",
    "os_version_id": "8.3",
    "process_name": "ceph-mon",
    "stack_sig":
"957c21d558d0cba4cee9e8aaf9227b3b1b09738b8a4d2c9f4dc26d9233b0d511",
    "timestamp": "2021-05-24T19:58:42.549073Z",
    "utsname_hostname": "host02",
    "utsname_machine": "x86_64",
```

```
    "utsname_release": "4.18.0-240.15.1.el8_3.x86_64",
    "utsname_sysname": "Linux",
    "utsname_version": "#1 SMP Wed Feb 3 03:12:15 EST 2021"
}
```

8. Remove saved crashes older than *KEEP* days: Here, *KEEP* must be an integer.

   Syntax

   ```
   ceph crash prune KEEP
   ```

   Example

   ```
   [root@mon ~]# ceph crash prune 60
   ```

9. Archive a crash report so that it is no longer considered for the **RECENT_CRASH** health check and does not appear in the **crash ls-new** output. It appears in the **crash ls**.

   Syntax

   ```
   ceph crash archive CRASH_ID
   ```

   Example

   ```
   [root@mon ~]# ceph crash archive 2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
   ```

10. Archive all crash reports:

    Example

    ```
    [root@mon ~]# ceph crash archive-all
    ```

11. Remove the crash dump:

    Syntax

    ```
    ceph crash rm CRASH_ID
    ```

    Example

    ```
    [root@mon ~]# ceph crash rm 2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
    ```

Additional Resources

- See the *Health messages of a Ceph cluster* section in the *Red Hat Ceph Storage Troubleshooting Guide* for more information on Ceph health messages.

1.11. MIGRATING RBD MIRRORING DAEMONS

For two-way Block device (RBD) mirroring configured using the command-line interface in a bare-metal storage cluster, the cluster does not migrate RBD mirroring. Migrate RBD mirror daemons from CLI to Ceph-Ansible prior to upgrading the storage cluster or converting the cluster to containerized.

Prerequisites

- A running Red Hat Ceph Storage non-containerized, bare-metal, cluster.
- Access to the Ansible administration node.
- An ansible user account.
- Sudo access to the ansible user account.

Procedure

1. Create a user on the Ceph client node:

   Syntax

   ```
   ceph auth get client.PRIMARY_CLUSTER_NAME -o
   /etc/ceph/ceph.PRIMARY_CLUSTER_NAME.keyring
   ```

   Example

   ```
   [root@rbd-client-site-a ~]# ceph auth get client.rbd-mirror.site-a -o /etc/ceph/ceph.client.rbd-
   mirror.site-a.keyring
   ```

2. Change the username in the **auth** file in **/etc/ceph** directory:

   Example

   ```
   [client.rbd-mirror.rbd-client-site-a]
       key = AQCbKbVg+E7POBAA7COSZCodvOrg2LWIFc9+3g==
       caps mds = "allow *"
       caps mgr = "allow *"
       caps mon = "allow *"
       caps osd = "allow *"
   ```

3. Import the **auth** file to add relevant permissions:

   Syntax

   ```
   ceph auth import -i PATH_TO_KEYRING
   ```

   Example

   ```
   [root@rbd-client-site-a ~]# ceph auth import -i /etc/ceph/ceph.client.rbd-mirror.rbd-client-site-
   a.keyring
   ```

4. Check the service name of the RBD mirror node:

   Example

```
[root@rbd-client-site-a ~]# systemctl list-units --all

systemctl stop ceph-rbd-mirror@rbd-client-site-a.service
systemctl disable ceph-rbd-mirror@rbd-client-site-a.service
systemctl reset-failed ceph-rbd-mirror@rbd-client-site-a.service
systemctl start ceph-rbd-mirror@rbd-mirror.rbd-client-site-a.service
systemctl enable ceph-rbd-mirror@rbd-mirror.rbd-client-site-a.service
systemctl status ceph-rbd-mirror@rbd-mirror.rbd-client-site-a.service
```

5. Add the rbd-mirror node to the **/etc/ansible/hosts** file:

Example

```
[rbdmirrors]
ceph.client.rbd-mirror.rbd-client-site-a
```

1.12. ADDITIONAL RESOURCES

- See the *Red Hat Ceph Storage Installation Guide* for details on installing the Red Hat Ceph Storage product.
- See the *Placement Groups (PGs)* chapter in the Red Hat Ceph Storage Strategies Guide for more information.
- See the *Red Hat Enterprise Linux 8 Configuring and Managing Logical Volumes* guide for more details.

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.

2.3. 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.

Prerequisites

- A healthy, and running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph OSD node.

Procedure

1. Remove the block device from **sysfs**:

Syntax

```
echo 1 > /sys/block/BLOCK_DEVICE/device/delete
```

Example

```
[root@osd ~]# 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@osd ~]# tail -50 /var/log/ceph/ceph-osd.1.log
2020-09-02 15:50:50.187067 7ff1ce9a8d80  1 bdev(0x563d263d4600 /var/lib/ceph/osd/ceph-2/block) close
2020-09-02 15:50:50.440398 7ff1ce9a8d80 -1 osd.2 0 OSD:init: unable to mount object store
2020-09-02 15:50:50.440416 7ff1ce9a8d80 -1 ^[[0;31m ** ERROR: osd init failed: (5) Input/output error^[[0m
2020-09-02 15:51:10.633738 7f495c44bd80  0 set uid:gid to 167:167 (ceph:ceph)
2020-09-02 15:51:10.633752 7f495c44bd80  0 ceph version 12.2.12-124.el7cp (e8948288b90d312c206301a9fcf80788fbc3b1f8) luminous (stable), process ceph-osd, pid 36209
2020-09-02 15:51:10.634703 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.635749 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.636642 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.637535 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.641256 7f495c44bd80  0 pidfile_write: ignore empty --pid-file
2020-09-02 15:51:10.669317 7f495c44bd80  0 load: jerasure load: lrc load: isa
2020-09-02 15:51:10.669387 7f495c44bd80  1 bdev create path /var/lib/ceph/osd/ceph-2/block type kernel
2020-09-02 15:51:10.669395 7f495c44bd80  1 bdev(0x55a423da9200 /var/lib/ceph/osd/ceph-2/block) open path /var/lib/ceph/osd/ceph-2/block
2020-09-02 15:51:10.669611 7f495c44bd80  1 bdev(0x55a423da9200 /var/lib/ceph/osd/ceph-2/block) open size 500103643136 (0x7470800000, 466GiB) block_size
```

```
4096 (4KiB) rotational
2020-09-02 15:51:10.670320 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.670328 7f495c44bd80  1 bdev(0x55a423da9200
/var/lib/ceph/osd/ceph-2/block) close
2020-09-02 15:51:10.924727 7f495c44bd80  1 bluestore(/var/lib/ceph/osd/ceph-2) _mount
path /var/lib/ceph/osd/ceph-2
2020-09-02 15:51:10.925582 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.925628 7f495c44bd80  1 bdev create path /var/lib/ceph/osd/ceph-
2/block type kernel
2020-09-02 15:51:10.925630 7f495c44bd80  1 bdev(0x55a423da8600
/var/lib/ceph/osd/ceph-2/block) open path /var/lib/ceph/osd/ceph-2/block
2020-09-02 15:51:10.925784 7f495c44bd80  1 bdev(0x55a423da8600
/var/lib/ceph/osd/ceph-2/block) open size 500103643136 (0x7470800000, 466GiB) block_size
4096 (4KiB) rotational
2020-09-02 15:51:10.926549 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
```

2. Looking at Ceph OSD disk tree, we also see the disk is offline.

Example

```
[root@osd ~]# 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
```

2.4. 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.

Prerequisites

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

Procedure

1. Check storage cluster health:

```
[root@mon ~]# ceph health
```

2. Identify the OSD location in the CRUSH hierarchy:

```
[root@mon ~]# ceph osd tree | grep -i down
```

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

   Syntax

   ```
   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 with the *OSD_ID*:

   Syntax

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

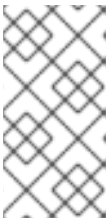
   If the command indicates that the OSD is already running, there might be a heartbeat or networking issue. If you cannot restart the OSD, then the drive might have failed.

   NOTE

   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.

   ```
   [root@osd ~]# 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:

   Syntax

   ```
   yum install smartmontools
   smartctl -H /dev/BLOCK_DEVICE
   ```

Example

```
[root@osd ~]# smartctl -H /dev/sda
```

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

6. Stop the OSD process:

   Syntax

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

7. For containerized deployments of Ceph, stop the OSD container:

   Syntax

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

8. Remove the OSD out of the storage cluster:

   Syntax

   ```
   ceph osd out OSD_ID
   ```

9. Ensure the failed OSD is backfilling:

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

10. Remove the OSD from the CRUSH Map:

    Syntax

    ```
    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:

    Syntax

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

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

    Example

    ```
    [root@osd ~]# ceph auth list
    ```

13. Remove the OSD from the storage cluster:

    Syntax

    ```
    ceph osd rm osd.OSD_ID
    ```

14. Unmount the failed drive path:

    Syntax

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

    Example

    ```
    [root@osd ~]# umount /var/lib/ceph/osd/ceph-0
    ```

    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.

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:

    Example

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

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

    Example

    ```
    [root@osd ~]# 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 Ceph Ansible.

    b. Using the command-line interface.

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

    Example

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

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

Syntax

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

19. Verify the OSD is online.

2.5. 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.

Prerequisites

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

Procedure

1. Destroy the OSD:

   Syntax

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

   Example

   ```
   [root@osd ~]# 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:

   Syntax

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

   Example

   ```
   [root@osd ~]# ceph-volume lvm zap /dev/sdb
   ```

   NOTE

   You can find the *DEVICE* by comparing output from various commands, such as **ceph osd tree**, **ceph osd metadata**, and **df**.

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

   Syntax

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

   Example

   ```
   [root@mon ~]# ceph-volume lvm create --osd-id 1 --data /dev/sdb
   ```

Additional Resources

- See the *Adding a Ceph OSD using Ansible with the same disk topologies* section in the *Red Hat Ceph Storage Operations Guide* for more details.
- See the *Adding a Ceph OSD using Ansible with different disk topologies* section in the *Red Hat Ceph Storage Operations Guide* for more details.
- See the *Preparing Ceph OSDs using `ceph-volume`* section in the *Red Hat Ceph Storage Operations Guide* for more details.
- See the *Activating Ceph OSDs using `ceph-volume`* section in the *Red Hat Ceph Storage Operations Guide* for more details.
- See the *Adding a Ceph OSD using the command-line interface* section in the *Red Hat Ceph Storage Operations Guide* for more details.

CHAPTER 3. HANDLING A NODE FAILURE

As a storage administrator, you can 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 placement groups (PGs) 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 that you can resize the storage cluster capacity or replace hardware without taking down the storage cluster.

The ability to serve Ceph clients while the storage 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 on weekends. However, adding and removing Ceph OSD nodes can have a significant impact on performance.

Before you add or remove Ceph OSD nodes, consider the effects on storage cluster performance:

- Whether you are expanding or reducing the storage cluster capacity, adding or removing Ceph OSD nodes induces backfilling as the storage cluster rebalances. During that rebalancing time period, Ceph uses additional resources, which can impact storage cluster performance.
- In a production Ceph storage cluster, a Ceph OSD node has a particular hardware configuration that facilitates a particular type of storage strategy.
- 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 the CRUSH ruleset.

Additional Resources

- *Red Hat Ceph Storage Storage Strategies Guide*

3.3. PERFORMANCE CONSIDERATIONS

The following factors typically affect a storage cluster's performance when adding or removing Ceph OSD nodes:

- Ceph clients place load on the I/O interface to Ceph; that is, the clients place 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 that is experiencing high client load, the client load could significantly affect recovery time and reduce performance. Because write operations require data replication for durability, write-intensive client loads in particular can increase the time for the storage cluster to recover.
- Generally, the capacity you are adding or removing affects the storage cluster's time to recover. In addition, the storage density of the node you add or remove might also affect recovery times. For example, a node with 36 OSDs typically takes longer to recover than a node with 12 OSDs.
- When removing nodes, you MUST ensure that you have sufficient spare capacity so that you will not reach **full ratio** or **near full ratio**. If the storage cluster reaches **full ratio**, Ceph will suspend write operations to prevent data loss.
- 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 a CRUSH ruleset experiences a performance impact when Ceph OSD nodes are added or removed.

- 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 that exist of the data, the longer it takes for the storage cluster to recover. For example, a larger pool or one that has a greater number of **k+m** chunks will take longer to recover than a replication pool with fewer copies of the same data.

- Drives, controllers and network interface cards all have throughput characteristics that might impact the recovery time. Generally, nodes with higher throughput characteristics, such as 10 Gbps and SSDs, recover more quickly than nodes with lower throughput characteristics, such as 1 Gbps and SATA drives.

3.4. RECOMMENDATIONS FOR ADDING OR REMOVING NODES

Red Hat recommends adding or removing one OSD at a time within a node and allowing the storage cluster to recover before proceeding to the next OSD. This helps to minimize the impact on storage cluster performance. Note that if a node fails, you might need to change the entire node at once, rather than one OSD at a time.

To remove an OSD:

- Using Ansible.

- Using the command-line interface.

To add an OSD:

- Using Ansible.

- Using the command-line interface.

When adding or removing Ceph OSD nodes, consider that other ongoing processes also affect storage cluster performance. 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 storage 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.

```
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, there is nothing wrong with operating in a **degraded** state. For example, you can operate the storage cluster with **osd_pool_default_size = 3** and **osd_pool_default_min_size = 2**. You can tune the storage cluster for the fastest possible recovery

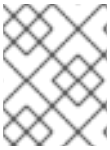
time, but doing so significantly affects Ceph client I/O performance. To maintain the highest Ceph client I/O performance, limit the backfill and recovery operations and allow them to take longer.

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

You can also consider setting the sleep and delay parameters such as, **osd_recovery_sleep**.

Increase the number of placement groups

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 amount will cause a considerable degradation in performance.



NOTE

See the KnowledgeBase article *How do I increase placement group (PG) count in a Ceph Cluster* for additional details.

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 8.
- Review the *Requirements for Installing Red Hat Ceph Storage* chapter in the *Red Hat Ceph Storage Installation Guide*.

Procedure

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

   Example

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

3. Limit the backfill and recovery features:

   Syntax

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

   Example

```
[root@mon ~]# 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@mon ~]# 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.

6. Enable scrubbing:

   Syntax

   ```
   ceph osd unset noscrub
   ceph osd unset nodeep-scrub
   ```

7. Set the backfill and recovery features to default:

   Syntax

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

   Example

   ```
   [root@mon ~]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active 3 --
   osd-recovery-op-priority 3
   ```

Additional Resources

- See the *Setting a Specific Configuration Setting at Runtime* section in the *Red Hat Ceph Storage Configuration Guide* for more details.
- See *Adding a Bucket* and *Moving 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 storage cluster to refuse write operations.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to all nodes in the storage cluster.

Procedure

1. Check the storage cluster's capacity:

   Syntax

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

2. Temporarily disable scrubbing:

   Syntax

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

3. Limit the backfill and recovery features:

   Syntax

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

   Example

   ```
   [root@mon ~]# 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 remove the next OSD.

a. After you remove an OSD, check to verify that the storage cluster is not getting to the **near-full ratio**:

Syntax

```
ceph -s
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@mon ~]# ceph osd crush rm node2
```

6. Enable scrubbing:

Syntax

```
ceph osd unset noscrub
ceph osd unset nodeep-scrub
```

7. Set the backfill and recovery features to default:

Syntax

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

Example

```
[root@mon ~]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active 3 --
osd-recovery-op-priority 3
```

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 a hard node failure, power off the node and reinstall the operating system.

Prerequisites

- A healthy running Red Hat Ceph Storage cluster.
- Root-level access to all nodes on the storage cluster.

Procedure

1. Check the storage cluster's capacity to understand the impact of removing the node:

   Example

   ```
   [root@ceph1 ~]# ceph df
   [root@ceph1 ~]# rados df
   [root@ceph1 ~]# ceph osd df
   ```

2. Optionally, disable recovery and backfilling:

   Example

   ```
   [root@ceph1 ~]# ceph osd set noout
   [root@ceph1 ~]# ceph osd set noscrub
   [root@ceph1 ~]# ceph osd set nodeep-scrub
   ```

3. Shut down the node.

4. If you are changing the host name, remove the node from CRUSH map:

   Example

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

5. Check the status of the storage cluster:

   Example

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

6. Reinstall the operating system on the node.

7. Add an Ansible user and generate the SSH keys:

   Example

   ```
   [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 Ansible administration node, copy the SSH keys for the **ansible** user on the reinstalled node:

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

9. From the Ansible administration node, run the Ansible playbook again:

Example

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

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:

Example

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

11. Check Ceph's health:

Example

```
[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

- The *Red Hat Ceph Storage Installation Guide*.
- See the *Configuring Ansible's inventory location* section in the *{storage_product} Installation Guide* for more details on the Ansible inventory configuration.

CHAPTER 4. HANDLING A DATA CENTER FAILURE

As a storage administrator, you can take preventive measures to avoid a data center failure. These preventive measures include:

- Configuring the data center infrastructure.
- Setting up failure domains within the CRUSH map hierarchy.
- Designating failure nodes within the domains.

4.1. PREREQUISITES

- A healthy running Red Hat Ceph Storage cluster.
- Root-level access to all nodes in the storage cluster.

4.2. AVOIDING A DATA CENTER FAILURE

Configuring the data center infrastructure

Each data center within a stretch cluster can have a different storage cluster configuration to reflect local capabilities and dependencies. Set up replication between the data centers to help preserve the data. If one data center fails, the other data centers in the storage cluster contain copies of the data.

Setting up failure domains within the CRUSH map hierarchy

Failure, or failover, domains are redundant copies of domains within the storage cluster. If an active domain fails, the failure domain becomes the active domain.

By default, the CRUSH map lists all nodes in a storage cluster within a flat hierarchy. However, for best results, create a logical hierarchical structure within the CRUSH map. The hierarchy designates the domains to which each node belongs and the relationships among those domains within the storage cluster, including the failure domains. Defining the failure domains for each domain within the hierarchy improves the reliability of the storage cluster.

When planning a storage cluster that contains multiple data centers, place the nodes within the CRUSH map hierarchy so that if one data center goes down, the rest of the storage cluster stays up and running.

Designating failure nodes within the domains

If you plan to use three-way replication for data within the storage cluster, consider the location of the nodes within the failure domain. If an outage occurs within a data center, it is possible that some data might reside in only 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. However, if some data exist in only one copy, the storage cluster reverts to read-only mode.

4.3. HANDLING A DATA CENTER FAILURE

Red Hat Ceph Storage can withstand catastrophic failures to the infrastructure, such as losing one of the 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 storage 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 a three data center environment, such as 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 example below, the resulting map is derived from the initial setup of the storage 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.

Example

```
[root@mon ~]# 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:

Example

```
[root@mon ~]# 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 are not available. Those misplacements will be corrected automatically once it is possible to do so.

Additional Resources

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

CHAPTER 5. MIGRATING A NON-CONTAINERIZED RED HAT CEPH STORAGE CLUSTER TO A CONTAINERIZED ENVIRONMENT

To manually migrate a non-containerized, bare-metal, Red Hat Ceph Storage cluster to a containerized environment, use the ceph-ansible **switch-from-non-containerized-to-containerized-ceph-daemons.yml** playbook.

NOTE

If the storage cluster has an RBD mirror daemon not deployed by **ceph-ansible**, you need to migrate the daemons prior to converting to a containerized cluster. For more details, see *Migrating RBD mirroring daemons*.

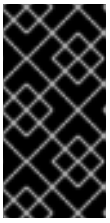
Prerequisites

- A running Red Hat Ceph Storage non-containerized, bare-metal, cluster.
- Access to the Ansible administration node.
- An ansible user account.
- Sudo access to the ansible user account.

Procedure

1. Edit the **group_vars/all.yml** file to include configuration for containers:

```
ceph_docker_image_tag: "latest"
ceph_docker_image: rhceph/rhceph-4-rhel8
containerized_deployment: true
ceph_docker_registry: registry.redhat.io
```



IMPORTANT

For the **ceph_docker_image_tag**, use **latest** if your current storage cluster is on latest version or use the appropriate image tag. See the *What are the Red Hat Ceph Storage releases and corresponding Ceph package versions?* for more information.

2. Navigate to the **/usr/share/ceph-ansible** directory:

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

3. On the Ansible administration node, run the Ansible migration playbook:

   Syntax

```
ansible-playbook ./infrastructure-playbooks/switch-from-non-containerized-to-containerized-ceph-daemons.yml -i INVENTORY_FILE
```

   Example

```
[ansible@admin ceph-ansible]$ ansible-playbook ./infrastructure-playbooks/switch-from-non-containerized-to-containerized-ceph-daemons.yml -i hosts
```

Verify the cluster is switched to containerized environment.

4. On the monitor node, list all running containers:

Red Hat Enterprise Linux 7

```
[root@mon ~]$ sudo docker ps
```

Red Hat Enterprise Linux 8

```
[root@mon ~]$ sudo podman ps
```

Additional Resources

- See the *Installing a Red Hat Ceph Storage cluster* chapter in the *Red Hat Ceph Storage Installation Guide* for information on installation of a bare-metal storage cluster.
- See the *Creating an Ansible user with sudo access* section in the *Red Hat Ceph Storage Installation Guide* for providing **sudo** access to the ansible user.
- See the *Configuring two-way mirroring using the command-line interface* section in the *Red Hat Ceph Storage Block Device Guide* for more details.