Red Hat Ceph Storage 4

Troubleshooting Guide

Troubleshooting Red Hat Ceph Storage

Last Updated: 2023-07-25

Red Hat Ceph Storage 4 Troubleshooting Guide

Troubleshooting Red Hat Ceph Storage

Legal Notice

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document describes how to resolve common problems with 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. INITIAL TROUBLESHOOTING

This chapter includes information on:

- How to start troubleshooting Ceph errors (Identifying problems)
- Most common **ceph health** error messages (Understanding Ceph Health)
- Most common Ceph log error messages (Understanding Ceph log)

1.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.

1.2. IDENTIFYING PROBLEMS

To determine possible causes of the error with the Red Hat Ceph Storage cluster, answer the questions in the Procedure section.

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. Certain problems can arise when using unsupported configurations. Ensure that your configuration is supported.
2. Do you know what Ceph component causes the problem?
    a. No. Follow *Diagnosing the health of a Ceph storage cluster* procedure in the *Red Hat Ceph Storage Troubleshooting Guide*.
    b. Ceph Monitors. See *Troubleshooting Ceph Monitors* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
    c. Ceph OSDs. See *Troubleshooting Ceph OSDs* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
    d. Ceph placement groups. See *Troubleshooting Ceph placement groups* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
    e. Multi-site Ceph Object Gateway. See *Troubleshooting a multi-site Ceph Object Gateway* section in the *Red Hat Ceph Storage Troubleshooting Guide*.

Additional Resources

- See the Red Hat Ceph Storage: Supported configurations article for details.

1.2.1. Diagnosing the health of a storage cluster

This procedure lists basic steps to diagnose the health of a Red Hat Ceph Storage cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. Check the overall status of the storage cluster:

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

   If the command returns **HEALTH_WARN** or **HEALTH_ERR** see *Understanding Ceph health* for details.

2. Check the Ceph logs for any error messages listed in *Understanding Ceph logs*. The logs are located by default in the **/var/log/ceph/** directory.

3. If the logs do not include sufficient amount of information, increase the debugging level and try to reproduce the action that failed. See *Configuring logging* for details.

1.3. UNDERSTANDING CEPH HEALTH

The **ceph health** command returns information about the status of the Red Hat Ceph Storage cluster:

- **HEALTH_OK** indicates that the cluster is healthy.
- **HEALTH_WARN** indicates a warning. In some cases, the Ceph status returns to **HEALTH_OK** automatically. For example when Red Hat Ceph Storage cluster finishes the rebalancing process. However, consider further troubleshooting if a cluster is in the **HEALTH_WARN** state for longer time.
- **HEALTH_ERR** indicates a more serious problem that requires your immediate attention.

Use the **ceph health detail** and **ceph -s** commands to get a more detailed output.

Additional Resources

- See the *Ceph Monitor error messages* table in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Ceph OSD error messages* table in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Placement group error messages* table in the *Red Hat Ceph Storage Troubleshooting Guide*.

1.4. UNDERSTANDING CEPH LOGS

1.4.1. Non containerized deployment

By default, Ceph stores its logs in the **/var/log/ceph/** directory.

The ***CLUSTER_NAME*.log** is the main storage cluster log file that includes global events. By default, the log file name is **ceph.log**. Only the Ceph Monitor nodes include the main storage cluster log.

Each Ceph OSD and Monitor has its own log file, named ***CLUSTER_NAME*-osd.*NUMBER*.log** and ***CLUSTER_NAME*-mon.*HOSTNAME*.log**.

When you increase debugging level for Ceph subsystems, Ceph generates new log files for those subsystems as well.

1.4.2. Container-based deployment

For container-based deployment, by default, Ceph log to **journald**, accessible using the **journactl** command. However, you can configure Ceph to log to files in **/var/log/ceph** in the configuration settings.

1. To enable logging Ceph Monitors, Ceph Manager, Ceph Object Gateway, and any other daemons, set **log_to_file** to **true** under [global] settings.

   Example

   ```
   [ceph: root@host01 ~]# ceph config set global log_to_file true
   ```

2. To enable logging for Ceph Monitor cluster and audit logs, set **mon_cluster_log_to_file** to **true**.

   Example

   ```
   [ceph: root@host01 ~]# ceph config set mon mon_cluster_log_to_file true
   ```

NOTE

If you choose to log to files, it is recommended to disable logging to **journald** or else everything is logged twice. Run the following commands to disable logging to **journald**:

```
# ceph config set global log_to_journald false
# ceph config set global mon_cluster_log_to_journald false
```

Additional Resources

- For details about logging, see *Configuring logging* in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Common Ceph Monitor error messages in the Ceph logs* table in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Common Ceph OSD error messages in the Ceph logs* table in the *Red Hat Ceph Storage Troubleshooting Guide*.

1.5. GATHERING LOGS FROM MULTIPLE HOSTS IN A CEPH CLUSTER USING ANSIBLE

Starting with Red Hat Ceph Storage 4.2, you can use **ceph-ansible** to gather logs from multiple hosts in a Ceph cluster. It captures **etc/ceph** and **/var/log/ceph** directories from the Ceph nodes. This playbook can be used to collect logs for a bare-metal and containerized storage cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster.

- Root-level access to the nodes.
- The **ceph-ansible** package is installed on the node.

Procedure

1. Log into the Ansible administration node as an ansible user.

NOTE

Ensure the node has adequate space to collect the logs from the hosts.

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

   Example

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

3. Run the Ansible playbook to gather the logs:

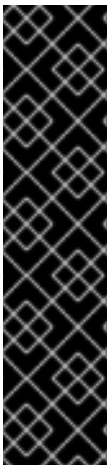
   Example

   ```
   ansible@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/gather-ceph-logs.yml -i hosts
   ```

   The logs are stored in the **/tmp** directory of the Ansible node.

CHAPTER 2. CONFIGURING LOGGING

This chapter describes how to configure logging for various Ceph subsystems.



IMPORTANT

Logging is resource intensive. Also, verbose logging can generate a huge amount of data in a relatively short time. If you are encountering problems in a specific subsystem of the cluster, enable logging only of that subsystem. See Section 2.2, "Ceph subsystems" for more information.

In addition, consider setting up a rotation of log files. See Section 2.5, "Accelerating log rotation" for details.

Once you fix any problems you encounter, change the subsystems log and memory levels to their default values. See Appendix A, *Ceph subsystems default logging level values* for list of all Ceph subsystems and their default values.

You can configure Ceph logging by:

- Using the **ceph** command at runtime. This is the most common approach. See Section 2.3, "Configuring logging at runtime" for details.
- Updating the Ceph configuration file. Use this approach if you are encountering problems when starting the cluster. See Section 2.4, "Configuring logging in configuration file" for details.

2.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.

2.2. CEPH SUBSYSTEMS

This section contains information about Ceph subsystems and their logging levels.

Understanding Ceph Subsystems and Their Logging Levels

Ceph consists of several subsystems.

Each subsystem has a logging level of its:

- Output logs that are stored by default in **/var/log/ceph/** directory (log level)
- Logs that are stored in a memory cache (memory level)

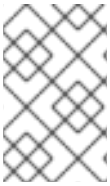
In general, Ceph does not send logs stored in memory to the output logs unless:

- A fatal signal is raised
- An assert in source code is triggered
- You request it

You can set different values for each of these subsystems. Ceph logging levels operate on scale of **1** to **20**, where **1** is terse and **20** is verbose.

Use a single value for the log level and memory level to set them both to the same value. For example, **debug_osd = 5** sets the debug level for the **ceph-osd** daemon to **5**.

To use different values for the output log level and the memory level, separate the values with a forward slash (/). For example, **debug_mon = 1/5** sets the debug log level for the **ceph-mon** daemon to **1** and its memory log level to **5**.



NOTE

For container-based deployment, Ceph generates logs to **journald**. You can enable logging to files in **/var/log/ceph** by setting **log_to_file** parameter to **true** under [global] in the Ceph configuration file. See *Understanding ceph logs* for more details.

Table 2.1. Ceph Subsystems and the Logging Default Values

Subsystem	Log Level	Memory Level	Description
asok	1	5	The administration socket
auth	1	5	Authentication
client	0	5	Any application or library that uses **librados** to connect to the cluster
bluestore	1	5	The BlueStore OSD backend
journal	1	5	The OSD journal
mds	1	5	The Metadata Servers
monc	0	5	The Monitor client handles communication between most Ceph daemons and Monitors
mon	1	5	Monitors
ms	0	5	The messaging system between Ceph components
osd	0	5	The OSD Daemons
paxos	0	5	The algorithm that Monitors use to establish a consensus
rados	0	5	Reliable Autonomic Distributed Object Store, a core component of Ceph
rbd	0	5	The Ceph Block Devices
rgw	1	5	The Ceph Object Gateway

Example Log Outputs

The following examples show the type of messages in the logs when you increase the verbosity for the Monitors and OSDs.

Monitor Debug Settings

```
debug_ms = 5
debug_mon = 20
debug_paxos = 20
debug_auth = 20
```

Example Log Output of Monitor Debug Settings

```
2016-02-12 12:37:04.278761 7f45a9afc700 10 mon.cephn2@0(leader).osd e322 e322: 2 osds: 2 up,
2 in
2016-02-12 12:37:04.278792 7f45a9afc700 10 mon.cephn2@0(leader).osd e322
min_last_epoch_clean 322
2016-02-12 12:37:04.278795 7f45a9afc700 10 mon.cephn2@0(leader).log v1010106 log
2016-02-12 12:37:04.278799 7f45a9afc700 10 mon.cephn2@0(leader).auth v2877 auth
2016-02-12 12:37:04.278811 7f45a9afc700 20 mon.cephn2@0(leader) e1 sync_trim_providers
2016-02-12 12:37:09.278914 7f45a9afc700 11 mon.cephn2@0(leader) e1 tick
2016-02-12 12:37:09.278949 7f45a9afc700 10 mon.cephn2@0(leader).pg v8126 v8126: 64 pgs: 64
active+clean; 60168 kB data, 172 MB used, 20285 MB / 20457 MB avail
2016-02-12 12:37:09.278975 7f45a9afc700 10 mon.cephn2@0(leader).paxosservice(pgmap
7511..8126) maybe_trim trim_to 7626 would only trim 115 < paxos_service_trim_min 250
2016-02-12 12:37:09.278982 7f45a9afc700 10 mon.cephn2@0(leader).osd e322 e322: 2 osds: 2 up,
2 in
2016-02-12 12:37:09.278989 7f45a9afc700  5 mon.cephn2@0(leader).paxos(paxos active c
1028850..1029466) is_readable = 1 - now=2016-02-12 12:37:09.278990 lease_expire=0.000000 has
v0 lc 1029466
....
2016-02-12 12:59:18.769963 7f45a92fb700  1 -- 192.168.0.112:6789/0 <== osd.1
192.168.0.114:6800/2801 5724 ==== pg_stats(0 pgs tid 3045 v 0) v1 ==== 124+0+0 (2380105412 0
0) 0x5d96300 con 0x4d5bf40
2016-02-12 12:59:18.770053 7f45a92fb700  1 -- 192.168.0.112:6789/0 --> 192.168.0.114:6800/2801
-- pg_stats_ack(0 pgs tid 3045) v1 -- ?+0 0x550ae00 con 0x4d5bf40
2016-02-12 12:59:32.916397 7f45a9afc700  0 mon.cephn2@0(leader).data_health(1) update_stats
avail 53% total 1951 MB, used 780 MB, avail 1053 MB
....
2016-02-12 13:01:05.256263 7f45a92fb700  1 -- 192.168.0.112:6789/0 --> 192.168.0.113:6800/2410
-- mon_subscribe_ack(300s) v1 -- ?+0 0x4f283c0 con 0x4d5b440
```

OSD Debug Settings

```
debug_ms = 5
debug_osd = 20
```

Example Log Output of OSD Debug Settings

```
2016-02-12 11:27:53.869151 7f5d55d84700  1 -- 192.168.17.3:0/2410 --> 192.168.17.4:6801/2801 --
osd_ping(ping e322 stamp 2016-02-12 11:27:53.869147) v2 -- ?+0 0x63baa00 con 0x578dee0
2016-02-12 11:27:53.869214 7f5d55d84700  1 -- 192.168.17.3:0/2410 --> 192.168.0.114:6801/2801
-- osd_ping(ping e322 stamp 2016-02-12 11:27:53.869147) v2 -- ?+0 0x638f200 con 0x578e040
```

```
2016-02-12 11:27:53.870215 7f5d6359f700  1 -- 192.168.17.3:0/2410 <== osd.1
192.168.0.114:6801/2801 109210 ==== osd_ping(ping_reply e322 stamp 2016-02-12
11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x63c1a00 con 0x578e040
2016-02-12 11:27:53.870698 7f5d6359f700  1 -- 192.168.17.3:0/2410 <== osd.1
192.168.17.4:6801/2801 109210 ==== osd_ping(ping_reply e322 stamp 2016-02-12
11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x6313200 con 0x578dee0
....
2016-02-12 11:28:10.432313 7f5d6e71f700  5 osd.0 322 tick
2016-02-12 11:28:10.432375 7f5d6e71f700 20 osd.0 322 scrub_random_backoff lost coin flip,
randomly backing off
2016-02-12 11:28:10.432381 7f5d6e71f700 10 osd.0 322 do_waiters -- start
2016-02-12 11:28:10.432383 7f5d6e71f700 10 osd.0 322 do_waiters -- finish
```

Additional Resources

- *Configuring logging at runtime*
- *Configuring logging in configuration file*
- *Understanding ceph logs*

2.3. CONFIGURING LOGGING AT RUNTIME

You can configure the logging of Ceph subsystems at system runtime to help troubleshoot any issues that might occur.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Access to Ceph debugger.

Procedure

1. To activate the Ceph debugging output, **dout()**, at runtime:

   ```
   ceph tell TYPE.ID injectargs --debug-SUBSYSTEM VALUE [--NAME VALUE]
   ```

2. Replace:

   - ***TYPE*** with the type of Ceph daemons (**osd**, **mon**, or **mds**)
   - ***ID*** with a specific ID of the Ceph daemon. Alternatively, use ***** to apply the runtime setting to all daemons of a particular type.
   - ***SUBSYSTEM*** with a specific subsystem.
   - ***VALUE*** with a number from **1** to **20**, where **1** is terse and **20** is verbose.
     For example, to set the log level for the OSD subsystem on the OSD named **osd.0** to 0 and the memory level to 5:

     ```
     # ceph tell osd.0 injectargs --debug-osd 0/5
     ```

To see the configuration settings at runtime:

1. Log in to the host with a running Ceph daemon, for example **ceph-osd** or **ceph-mon**.
2. Display the configuration:

```
ceph daemon NAME config show | less
```

Example

```
# ceph daemon osd.0 config show | less
```

Additional Resources

- See *Ceph subsystems* for details.
- See *Configuration logging in configuration file* for details.
- The *Ceph Debugging and Logging Configuration Reference* chapter in the *Configuration Guide* for Red Hat Ceph Storage 4.

2.4. CONFIGURING LOGGING IN CONFIGURATION FILE

Configure Ceph subsystems to log informational, warning, and error messages to the log file. You can specify the debugging level in the Ceph configuration file, by default **/etc/ceph/ceph.conf**.

Prerequisites

- A running Red Hat Ceph Storage cluster.

Procedure

1. To activate Ceph debugging output, **dout()** at boot time, add the debugging settings to the Ceph configuration file.
   a. For subsystems common to each daemon, add the settings under the **[global]** section.
   b. For subsystems for particular daemons, add the settings under a daemon section, such as **[mon]**, **[osd]**, or **[mds]**.

Example

```
[global]
        debug_ms = 1/5

[mon]
        debug_mon = 20
        debug_paxos = 1/5
        debug_auth = 2

[osd]
        debug_osd = 1/5
        debug_monc = 5/20

[mds]
        debug_mds = 1
```

Additional Resources

- *Ceph subsystems*
- *Configuring logging at runtime*
- The *Ceph Debugging and Logging Configuration Reference* chapter in the *Configuration Guide* for Red Hat Ceph Storage 4

2.5. ACCELERATING LOG ROTATION

Increasing debugging level for Ceph components might generate a huge amount of data. If you have almost full disks, you can accelerate log rotation by modifying the Ceph log rotation file at **/etc/logrotate.d/ceph**. The Cron job scheduler uses this file to schedule log rotation.

Prerequisites

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

Procedure

1. Add the size setting after the rotation frequency to the log rotation file:

```
rotate 7
weekly
size SIZE
compress
sharedscripts
```

   For example, to rotate a log file when it reaches 500 MB:

```
rotate 7
weekly
size 500 MB
compress
sharedscripts
size 500M
```

2. Open the **crontab** editor:

```
[root@mon ~]# crontab -e
```

3. Add an entry to check the **/etc/logrotate.d/ceph** file. For example, to instruct Cron to check **/etc/logrotate.d/ceph** every 30 minutes:

```
30 * * * * /usr/sbin/logrotate /etc/logrotate.d/ceph >/dev/null 2>&1
```

Additional Resources

- The *Scheduling a Recurring Job Using Cron* section in the System Administrator's Guide for Red Hat Enterprise Linux 7.

CHAPTER 3. TROUBLESHOOTING NETWORKING ISSUES

This chapter lists basic troubleshooting procedures connected with networking and Network Time Protocol (NTP).

3.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.

3.2. BASIC NETWORKING TROUBLESHOOTING

Red Hat Ceph Storage depends heavily on a reliable network connection. Red Hat Ceph Storage nodes use the network for communicating with each other. Networking issues can cause many problems with Ceph OSDs, such as them flapping, or being incorrectly reported as **down**. Networking issues can also cause the Ceph Monitor's clock skew errors. In addition, packet loss, high latency, or limited bandwidth can impact the cluster performance and stability.

Prerequisites

- Root-level access to the node.

Procedure

1. Installing the **net-tools** and **telnet** packages can help when troubleshooting network issues that can occur in a Ceph storage cluster:

   #### Red Hat Enterprise Linux 7

   ```
   [root@mon ~]# yum install net-tools
   [root@mon ~]# yum install telnet
   ```

   #### Red Hat Enterprise Linux 8

   ```
   [root@mon ~]# dnf install net-tools
   [root@mon ~]# dnf install telnet
   ```

2. Verify that the **cluster_network** and **public_network** parameters in the Ceph configuration file include the correct values:

   #### Example

   ```
   [root@mon ~]# cat /etc/ceph/ceph.conf | grep net
   cluster_network = 192.168.1.0/24
   public_network = 192.168.0.0/24
   ```

3. Verify that the network interfaces are up:

   #### Example

   ```
   [root@mon ~]# ip link list
   1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
   DEFAULT group default qlen 1000
   ```

```
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp22s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
mode DEFAULT group default qlen 1000
    link/ether 40:f2:e9:b8:a0:48 brd ff:ff:ff:ff:ff:ff
```

4. Verify that the Ceph nodes are able to reach each other using their short host names. Verify this on each node in the storage cluster:

   Syntax

   ```
   ping SHORT_HOST_NAME
   ```

   Example

   ```
   [root@mon ~]# ping osd01
   ```

5. If you use a firewall, ensure that Ceph nodes are able to reach other on their appropriate ports. The **firewall-cmd** and **telnet** tools can validate the port status, and if the port is open respectively:

   Syntax

   ```
   firewall-cmd --info-zone=ZONE
   telnet IP_ADDRESS PORT
   ```

   Example

   ```
   [root@mon ~]# firewall-cmd --info-zone=public
   public (active)
     target: default
     icmp-block-inversion: no
     interfaces: enp1s0
     sources: 192.168.0.0/24
     services: ceph ceph-mon cockpit dhcpv6-client ssh
     ports: 9100/tcp 8443/tcp 9283/tcp 3000/tcp 9092/tcp 9093/tcp 9094/tcp 9094/udp
     protocols:
     masquerade: no
     forward-ports:
     source-ports:
     icmp-blocks:
     rich rules:

   [root@mon ~]# telnet 192.168.0.22 9100
   ```

6. Verify that there are no errors on the interface counters. Verify that the network connectivity between nodes has expected latency, and that there is no packet loss.

   a. Using the **ethtool** command:

      Syntax

      ```
      ethtool -S INTERFACE
      ```

Example

```
[root@mon ~]# ethtool -S enp22s0f0 | grep errors
NIC statistics:
     rx_fcs_errors: 0
     rx_align_errors: 0
     rx_frame_too_long_errors: 0
     rx_in_length_errors: 0
     rx_out_length_errors: 0
     tx_mac_errors: 0
     tx_carrier_sense_errors: 0
     tx_errors: 0
     rx_errors: 0
```

b. Using the **ifconfig** command:

Example

```
[root@mon ~]# ifconfig
enp22s0f0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 10.8.222.13  netmask 255.255.254.0  broadcast 10.8.223.255
inet6 2620:52:0:8de:42f2:e9ff:feb8:a048  prefixlen 64  scopeid 0x0<global>
inet6 fe80::42f2:e9ff:feb8:a048  prefixlen 64  scopeid 0x20<link>
ether 40:f2:e9:b8:a0:48  txqueuelen 1000  (Ethernet)
RX packets 4219130  bytes 2704255777 (2.5 GiB)
RX errors 0  dropped 0  overruns 0  frame 0 (1)
TX packets 1418329  bytes 738664259 (704.4 MiB)
TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0 (2)
device interrupt 16
```

c. Using the **netstat** command:

Example

```
[root@mon ~]# netstat -ai
Kernel Interface table
Iface       MTU   RX-OK RX-ERR RX-DRP RX-OVR  TX-OK TX-ERR TX-DRP TX-OVR
Flg
docker0      1500      0      0      0 0           0      0      0      0 BMU
eno2         1500      0      0      0 0           0      0      0      0 BMU
eno3         1500      0      0      0 0           0      0      0      0 BMU
eno4         1500      0      0      0 0           0      0      0      0 BMU
enp0s20u13u5  1500  253277      0      0 0         0      0      0      0 BMRU
enp22s0f0    9000  234160      0      0 0    432326      0      0      0 BMRU (1)
lo          65536   10366      0      0 0     10366      0      0      0 LRU
```

7. For performance issues, in addition to the latency checks and to verify the network bandwidth between all nodes of the storage cluster, use the **iperf3** tool. The **iperf3** tool does a simple point-to-point network bandwidth test between a server and a client.

a. Install the **iperf3** package on the Red Hat Ceph Storage nodes you want to check the bandwidth:

Red Hat Enterprise Linux 7

```
[root@mon ~]# yum install iperf3
```

Red Hat Enterprise Linux 8

```
[root@mon ~]# dnf install iperf3
```

b. On a Red Hat Ceph Storage node, start the **iperf3** server:

Example

```
[root@mon ~]# iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

NOTE

The default port is 5201, but can be set using the **-P** command argument.

c. On a different Red Hat Ceph Storage node, start the **iperf3** client:

Example

```
[root@osd ~]# iperf3 -c mon
Connecting to host mon, port 5201
[  4] local xx.x.xxx.xx port 52270 connected to xx.x.xxx.xx port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec   114 MBytes   954 Mbits/sec    0    409 KBytes
[  4]   1.00-2.00   sec   113 MBytes   945 Mbits/sec    0    409 KBytes
[  4]   2.00-3.00   sec   112 MBytes   943 Mbits/sec    0    454 KBytes
[  4]   3.00-4.00   sec   112 MBytes   941 Mbits/sec    0    471 KBytes
[  4]   4.00-5.00   sec   112 MBytes   940 Mbits/sec    0    471 KBytes
[  4]   5.00-6.00   sec   113 MBytes   945 Mbits/sec    0    471 KBytes
[  4]   6.00-7.00   sec   112 MBytes   937 Mbits/sec    0    488 KBytes
[  4]   7.00-8.00   sec   113 MBytes   947 Mbits/sec    0    520 KBytes
[  4]   8.00-9.00   sec   112 MBytes   939 Mbits/sec    0    520 KBytes
[  4]   9.00-10.00  sec   112 MBytes   939 Mbits/sec    0    520 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  1.10 GBytes   943 Mbits/sec    0             sender
[  4]   0.00-10.00  sec  1.10 GBytes   941 Mbits/sec                  receiver

iperf Done.
```

This output shows a network bandwidth of 1.1 Gbits/second between the Red Hat Ceph Storage nodes, along with no retransmissions (**Retr**) during the test.

Red Hat recommends you validate the network bandwidth between all the nodes in the storage cluster.

8. Ensure that all nodes have the same network interconnect speed. Slower attached nodes might slow down the faster connected ones. Also, ensure that the inter switch links can handle the aggregated bandwidth of the attached nodes:

Syntax

```
ethtool INTERFACE
```

Example

```
[root@mon ~]# ethtool enp22s0f0
Settings for enp22s0f0:
Supported ports: [ TP ]
Supported link modes:   10baseT/Half 10baseT/Full
                        100baseT/Half 100baseT/Full
                        1000baseT/Half 1000baseT/Full
Supported pause frame use: No
Supports auto-negotiation: Yes
Supported FEC modes: Not reported
Advertised link modes:  10baseT/Half 10baseT/Full
                        100baseT/Half 100baseT/Full
                        1000baseT/Half 1000baseT/Full
Advertised pause frame use: Symmetric
Advertised auto-negotiation: Yes
Advertised FEC modes: Not reported
Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                     100baseT/Half 100baseT/Full
                                     1000baseT/Full
Link partner advertised pause frame use: Symmetric
Link partner advertised auto-negotiation: Yes
Link partner advertised FEC modes: Not reported
Speed: 1000Mb/s (1)
Duplex: Full (2)
Port: Twisted Pair
PHYAD: 1
Transceiver: internal
Auto-negotiation: on
MDI-X: off
Supports Wake-on: g
Wake-on: d
Current message level: 0x000000ff (255)
       drv probe link timer ifdown ifup rx_err tx_err
Link detected: yes (3)
```

Additional Resources

- See the *Basic Network troubleshooting* solution on the Customer Portal for details.
- See the *Verifying and configuring the MTU value* section in the *Red Hat Ceph Storage Configuration Guide*.
- See the *Configuring Firewall* section in the *Red Hat Ceph Storage Installation Guide*.
- See the *What is the "ethtool" command and how can I use it to obtain information about my network devices and interfaces* for details.
- See the *RHEL network interface dropping packets* solutions on the Customer Portal for details.

- For details, see the *What are the performance benchmarking tools available for Red Hat Ceph Storage?* solution on the Customer Portal.
- The *Networking Guide* for Red Hat Enterprise Linux 7.
- For more information, see *Knowledgebase articles and solutions* related to troubleshooting networking issues on the Customer Portal.

3.3. BASIC CHRONY NTP TROUBLESHOOTING

This section includes basic chrony troubleshooting steps.

Prerequisites

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

Procedure

1. Verify that the **chronyd** daemon is running on the Ceph Monitor hosts:

   Example

   ```
   [root@mon ~]# systemctl status chronyd
   ```

2. If **chronyd** is not running, enable and start it:

   Example

   ```
   [root@mon ~]# systemctl enable chronyd
   [root@mon ~]# systemctl start chronyd
   ```

3. Ensure that **chronyd** is synchronizing the clocks correctly:

   Example

   ```
   [root@mon ~]# chronyc sources
   [root@mon ~]# chronyc sourcestats
   [root@mon ~]# chronyc tracking
   ```

Additional Resources

- See the *How to troubleshoot chrony issues* solution on the Red Hat Customer Portal for advanced chrony NTP troubleshooting steps.
- See the *Clock skew* section in the *Red Hat Ceph Storage Troubleshooting Guide* for further details.
- See the *Checking if chrony is synchronized* section for further details.

3.4. BASIC NTP TROUBLESHOOTING

This section includes basic NTP troubleshooting steps.

Prerequisites

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

Procedure

1. Verify that the **ntpd** daemon is running on the Ceph Monitor hosts:

   #### Example

   ```
   [root@mon ~]# systemctl status ntpd
   ```

2. If **ntpd** is not running, enable and start it:

   #### Example

   ```
   [root@mon ~]# systemctl enable ntpd
   [root@mon ~]# systemctl start ntpd
   ```

3. Ensure that **ntpd** is synchronizing the clocks correctly:

   #### Example

   ```
   [root@mon ~]# ntpq -p
   ```

Additional Resources

- See the *How to troubleshoot NTP issues* solution on the Red Hat Customer Portal for advanced NTP troubleshooting steps.
- See the *Clock skew* section in the *Red Hat Ceph Storage Troubleshooting Guide* for further details.

CHAPTER 4. TROUBLESHOOTING CEPH MONITORS

This chapter contains information on how to fix the most common errors related to the Ceph Monitors.

4.1. PREREQUISITES

- Verify the network connection.

4.2. MOST COMMON CEPH MONITOR ERRORS

The following tables list the most common error messages that are returned by the **ceph health detail** command, or included in the Ceph logs. The tables provide links to corresponding sections that explain the errors and point to specific procedures to fix the problems.

4.2.1. Prerequisites

- A running Red Hat Ceph Storage cluster.

4.2.2. Ceph Monitor error messages

A table of common Ceph Monitor error messages, and a potential fix.

Error message	See
HEALTH_WARN	
mon.X is down (out of quorum)	Ceph Monitor is out of quorum
clock skew	Clock skew
store is getting too big!	The Ceph Monitor store is getting too big

4.2.3. Common Ceph Monitor error messages in the Ceph logs

A table of common Ceph Monitor error messages found in the Ceph logs, and a link to a potential fix.

Error message	Log file	See
clock skew	Main cluster log	Clock skew
clocks not synchronized	Main cluster log	Clock skew
Corruption: error in middle of record	Monitor log	Ceph Monitor is out of quorum Recovering the Ceph Monitor store

Error message	Log file	See
Corruption: 1 missing files	Monitor log	Ceph Monitor is out of quorum Recovering the Ceph Monitor store
Caught signal (Bus error)	Monitor log	Ceph Monitor is out of quorum

4.2.4. Ceph Monitor is out of quorum

One or more Ceph Monitors are marked as **down** but the other Ceph Monitors are still able to form a quorum. In addition, the **ceph health detail** command returns an error message similar to the following one:

```
HEALTH_WARN 1 mons down, quorum 1,2 mon.b,mon.c
mon.a (rank 0) addr 127.0.0.1:6789/0 is down (out of quorum)
```

What This Means

Ceph marks a Ceph Monitor as **down** due to various reasons.

If the **ceph-mon** daemon is not running, it might have a corrupted store or some other error is preventing the daemon from starting. Also, the **/var/** partition might be full. As a consequence, **ceph-mon** is not able to perform any operations to the store located by default at **/var/lib/ceph/mon-*SHORT_HOST_NAME*/store.db** and terminates.

If the **ceph-mon** daemon is running but the Ceph Monitor is out of quorum and marked as **down**, the cause of the problem depends on the Ceph Monitor state:

- If the Ceph Monitor is in the *probing* state longer than expected, it cannot find the other Ceph Monitors. This problem can be caused by networking issues, or the Ceph Monitor can have an outdated Ceph Monitor map (**monmap**) and be trying to reach the other Ceph Monitors on incorrect IP addresses. Alternatively, if the **monmap** is up-to-date, Ceph Monitor's clock might not be synchronized.
- If the Ceph Monitor is in the *electing* state longer than expected, the Ceph Monitor's clock might not be synchronized.
- If the Ceph Monitor changes its state from *synchronizing* to *electing* and back, the cluster state is advancing. This means that it is generating new maps faster than the synchronization process can handle.
- If the Ceph Monitor marks itself as the *leader* or a *peon*, then it believes to be in a quorum, while the remaining cluster is sure that it is not. This problem can be caused by failed clock synchronization.

To Troubleshoot This Problem

1. Verify that the **ceph-mon** daemon is running. If not, start it:

```
[root@mon ~]# systemctl status ceph-mon@HOST_NAME
[root@mon ~]# systemctl start ceph-mon@HOST_NAME
```

Replace ***HOST_NAME*** with the short name of the host where the daemon is running. Use the **hostname -s** command when unsure.

2. If you are not able to start **ceph-mon**, follow the steps in *The **ceph-mon** daemon cannot start*.

3. If you are able to start the **ceph-mon** daemon but is is marked as **down**, follow the steps in *The **ceph-mon** daemon is running, but marked as `down` .*

The ceph-mon Daemon Cannot Start

1. Check the corresponding Ceph Monitor log, by default located at **/var/log/ceph/ceph-mon.*HOST_NAME*.log**.

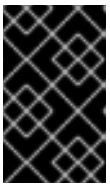
2. If the log contains error messages similar to the following ones, the Ceph Monitor might have a corrupted store.

```
Corruption: error in middle of record
Corruption: 1 missing files; e.g.: /var/lib/ceph/mon/mon.0/store.db/1234567.ldb
```

To fix this problem, replace the Ceph Monitor. See *Replacing a failed monitor* .

3. If the log contains an error message similar to the following one, the **/var/** partition might be full. Delete any unnecessary data from **/var/**.

```
Caught signal (Bus error)
```



IMPORTANT

Do not delete any data from the Monitor directory manually. Instead, use the **ceph-monstore-tool** to compact it. See *Compacting the Ceph Monitor store* for details.

4. If you see any other error messages, open a support ticket. See *Contacting Red Hat Support for service* for details.

The ceph-mon Daemon Is Running, but Still Marked as down

1. From the Ceph Monitor host that is out of the quorum, use the **mon_status** command to check its state:

```
[root@mon ~]# ceph daemon ID mon_status
```

Replace ***ID*** with the ID of the Ceph Monitor, for example:

```
[root@mon ~]# ceph daemon mon.a mon_status
```

2. If the status is *probing*, verify the locations of the other Ceph Monitors in the **mon_status** output.

a. If the addresses are incorrect, the Ceph Monitor has incorrect Ceph Monitor map (**monmap**). To fix this problem, see *Injecting a Ceph Monitor map* .

b. If the addresses are correct, verify that the Ceph Monitor clocks are synchronized. See *Clock skew* for details. In addition, troubleshoot any networking issues, see *Troubleshooting Networking issues* for details.

3. If the status is *electing*, verify that the Ceph Monitor clocks are synchronized. See *Clock skew* for details.

4. If the status changes from *electing* to *synchronizing*, open a support ticket. See *Contacting Red Hat Support for service* for details.

5. If the Ceph Monitor is the *leader* or a *peon*, verify that the Ceph Monitor clocks are synchronized. See *Clock skew* for details. Open a support ticket if synchronizing the clocks does not solve the problem. See *Contacting Red Hat Support for service* for details.

Additional Resources

- See *Understanding Ceph Monitor status*
- The *Starting, Stopping, Restarting the Ceph daemons by instance* section in the *Administration Guide* for Red Hat Ceph Storage 4
- The *Using the Ceph Administration Socket* section in the *Administration Guide* for Red Hat Ceph Storage 4

4.2.5. Clock skew

A Ceph Monitor is out of quorum, and the **ceph health detail** command output contains error messages similar to these:

```
mon.a (rank 0) addr 127.0.0.1:6789/0 is down (out of quorum)
mon.a addr 127.0.0.1:6789/0 clock skew 0.08235s > max 0.05s (latency 0.0045s)
```

In addition, Ceph logs contain error messages similar to these:

```
2015-06-04 07:28:32.035795 7f806062e700 0 log [WRN] : mon.a 127.0.0.1:6789/0 clock skew 0.14s
> max 0.05s
2015-06-04 04:31:25.773235 7f4997663700 0 log [WRN] : message from mon.1 was stamped
0.186257s in the future, clocks not synchronized
```

What This Means

The **clock skew** error message indicates that Ceph Monitors' clocks are not synchronized. Clock synchronization is important because Ceph Monitors depend on time precision and behave unpredictably if their clocks are not synchronized.

The **mon_clock_drift_allowed** parameter determines what disparity between the clocks is tolerated. By default, this parameter is set to 0.05 seconds.

IMPORTANT

Do not change the default value of **mon_clock_drift_allowed** without previous testing. Changing this value might affect the stability of the Ceph Monitors and the Ceph Storage Cluster in general.

Possible causes of the **clock skew** error include network problems or problems with Network Time Protocol (NTP) synchronization if that is configured. In addition, time synchronization does not work properly on Ceph Monitors deployed on virtual machines.

To Troubleshoot This Problem

1. Verify that your network works correctly. For details, see *Troubleshooting networking issues*. In particular, troubleshoot any problems with NTP clients if you use NTP.
   a. If you use chrony for NTP, see *Basic chrony NTP troubleshooting* section for more information.
   b. If you use **ntpd**, see *Basic NTP troubleshooting* .
2. If you use a remote NTP server, consider deploying your own NTP server on your network.
   a. For details, see the *Using the Chrony suite to configure NTP* chapter in the *Configuring basic system settings* for Red Hat Enterprise Linux 8.
   b. See the *Configuring NTP Using ntpd* chapter in the *System Administrator's Guide* for Red Hat Enterprise Linux 7.

NOTE

Ceph evaluates time synchronization every five minutes only so there will be a delay between fixing the problem and clearing the **clock skew** messages.

Additional Resources

- *Understanding Ceph Monitor status*
- *Ceph Monitor is out of quorum*

4.2.6. The Ceph Monitor store is getting too big

The **ceph health** command returns an error message similar to the following one:

```
mon.ceph1 store is getting too big! 48031 MB >= 15360 MB -- 62% avail
```

What This Means

Ceph Monitors store is in fact a LevelDB database that stores entries as key–values pairs. The database includes a cluster map and is located by default at **/var/lib/ceph/mon/*CLUSTER_NAME-SHORT_HOST_NAME*/store.db**.

Querying a large Monitor store can take time. As a consequence, the Ceph Monitor can be delayed in responding to client queries.

In addition, if the /**var**/ partition is full, the Ceph Monitor cannot perform any write operations to the store and terminates. See *Ceph Monitor is out of quorum* for details on troubleshooting this issue.

To Troubleshoot This Problem

1. Check the size of the database:

   ```
   du -sch /var/lib/ceph/mon/CLUSTER_NAME-SHORT_HOST_NAME/store.db
   ```

   Specify the name of the cluster and the short host name of the host where the **ceph-mon** is running.

   Example

   ```
   # du -sch /var/lib/ceph/mon/ceph-host1/store.db
   47G     /var/lib/ceph/mon/ceph-ceph1/store.db/
   47G     total
   ```

2. Compact the Ceph Monitor store. For details, see *Compacting the Ceph Monitor Store*.

Additional Resources

- Ceph Monitor is out of quorum

4.2.7. Understanding Ceph Monitor status

The **mon_status** command returns information about a Ceph Monitor, such as:

- State
- Rank
- Elections epoch
- Monitor map (**monmap**)

If Ceph Monitors are able to form a quorum, use **mon_status** with the **ceph** command-line utility.

If Ceph Monitors are not able to form a quorum, but the **ceph-mon** daemon is running, use the administration socket to execute **mon_status**.

An example output of mon_status

```
{
   "name": "mon.3",
   "rank": 2,
   "state": "peon",
   "election_epoch": 96,
   "quorum": [
      1,
      2
   ],
   "outside_quorum": [],
   "extra_probe_peers": [],
   "sync_provider": [],
```

```
    "monmap": {
        "epoch": 1,
        "fsid": "d5552d32-9d1d-436c-8db1-ab5fc2c63cd0",
        "modified": "0.000000",
        "created": "0.000000",
        "mons": [
            {
                "rank": 0,
                "name": "mon.1",
                "addr": "172.25.1.10:6789\/0"
            },
            {
                "rank": 1,
                "name": "mon.2",
                "addr": "172.25.1.12:6789\/0"
            },
            {
                "rank": 2,
                "name": "mon.3",
                "addr": "172.25.1.13:6789\/0"
            }
        ]
    }
}
```

Ceph Monitor States

Leader

During the electing phase, Ceph Monitors are electing a leader. The leader is the Ceph Monitor with the highest rank, that is the rank with the lowest value. In the example above, the leader is **mon.1**.

Peon

Peons are the Ceph Monitors in the quorum that are not leaders. If the leader fails, the peon with the highest rank becomes a new leader.

Probing

A Ceph Monitor is in the probing state if it is looking for other Ceph Monitors. For example after you start the Ceph Monitors, they are *probing* until they find enough Ceph Monitors specified in the Ceph Monitor map (**monmap**) to form a quorum.

Electing

A Ceph Monitor is in the electing state if it is in the process of electing the leader. Usually, this status changes quickly.

Synchronizing

A Ceph Monitor is in the synchronizing state if it is synchronizing with the other Ceph Monitors to join the quorum. The smaller the Ceph Monitor store it, the faster the synchronization process. Therefore, if you have a large store, synchronization takes longer time.

Additional Resources

- For details, see the *Using the Ceph Administration Socket* section in the *Administration Guide* for Red Hat Ceph Storage 4.

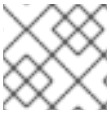
4.2.8. Additional Resources

- See the Section 4.2.2, "Ceph Monitor error messages" in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the Section 4.2.3, "Common Ceph Monitor error messages in the Ceph logs" in the *Red Hat Ceph Storage Troubleshooting Guide*.

4.3. INJECTING A MONMAP

If a Ceph Monitor has an outdated or corrupted Ceph Monitor map (**monmap**), it cannot join a quorum because it is trying to reach the other Ceph Monitors on incorrect IP addresses.

The safest way to fix this problem is to obtain and inject the actual Ceph Monitor map from other Ceph Monitors.



NOTE

This action overwrites the existing Ceph Monitor map kept by the Ceph Monitor.

This procedure shows how to inject the Ceph Monitor map when the other Ceph Monitors are able to form a quorum, or when at least one Ceph Monitor has a correct Ceph Monitor map. If all Ceph Monitors have corrupted store and therefore also the Ceph Monitor map, see *Recovering the Ceph Monitor store*.

Prerequisites

- Access to the Ceph Monitor Map.
- Root-level access to the Ceph Monitor node.

Procedure

1. If the remaining Ceph Monitors are able to form a quorum, get the Ceph Monitor map by using the **ceph mon getmap** command:

```
[root@mon ~]# ceph mon getmap -o /tmp/monmap
```

2. If the remaining Ceph Monitors are not able to form the quorum and you have at least one Ceph Monitor with a correct Ceph Monitor map, copy it from that Ceph Monitor:
   a. Stop the Ceph Monitor which you want to copy the Ceph Monitor map from:

```
[root@mon ~]# systemctl stop ceph-mon@<host-name>
```

   For example, to stop the Ceph Monitor running on a host with the **host1** short host name:

```
[root@mon ~]# systemctl stop ceph-mon@host1
```

   b. Copy the Ceph Monitor map:

```
[root@mon ~]# ceph-mon -i ID --extract-monmap /tmp/monmap
```

   Replace ***ID*** with the ID of the Ceph Monitor which you want to copy the Ceph Monitor map from:

```
[root@mon ~]# ceph-mon -i mon.a  --extract-monmap /tmp/monmap
```

3. Stop the Ceph Monitor with the corrupted or outdated Ceph Monitor map:

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

For example, to stop a Ceph Monitor running on a host with the **host2** short host name:

```
[root@mon ~]# systemctl stop ceph-mon@host2
```

4. You can inject the Ceph Monitor map as a **ceph** user in two different ways:

- Run the command as a **ceph** user:

Syntax

```
su - ceph -c 'ceph-mon -i ID --inject-monmap /tmp/monmap'
```

Replace ***ID*** with the ID of the Ceph Monitor with the corrupted or outdated Ceph Monitor map:

Example

```
[root@mon ~]# su - ceph -c 'ceph-mon -i mon.c --inject-monmap /tmp/monmap'
```

- Run the command as a **root** user and then run **chown** to change the permissions:

a. Run the command as a root user:

Syntax

```
ceph-mon -i ID --inject-monmap /tmp/monmap
```

Replace ***ID*** with the ID of the Ceph Monitor with the corrupted or outdated Ceph Monitor map:

Example

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

b. Change the file permissions:

Example

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

5. Start the Ceph Monitor:

```
[root@mon ~]# systemctl start ceph-mon@host2
```

If you copied the Ceph Monitor map from another Ceph Monitor, start that Ceph Monitor, too:

```
[root@mon ~]# systemctl start ceph-mon@host1
```

Additional Resources

- *Ceph Monitor is out of quorum*
- *Recovering the Ceph Monitor store*

4.4. REPLACING A FAILED MONITOR

When a Monitor has a corrupted store, the recommended way to fix this problem is to replace the Monitor by using the Ansible automation application.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Able to form a quroum.
- Root-level access to Ceph Monitor node.

Procedure

1. From the Monitor host, remove the Monitor store by default located at **/var/lib/ceph/mon/*CLUSTER_NAME-SHORT_HOST_NAME***:

   ```
   rm -rf /var/lib/ceph/mon/CLUSTER_NAME-SHORT_HOST_NAME
   ```

   Specify the short host name of the Monitor host and the cluster name. For example, to remove the Monitor store of a Monitor running on **host1** from a cluster called **remote**:

   ```
   [root@mon ~]# rm -rf /var/lib/ceph/mon/remote-host1
   ```

2. Remove the Monitor from the Monitor map (**monmap**):

   ```
   ceph mon remove SHORT_HOST_NAME --cluster CLUSTER_NAME
   ```

   Specify the short host name of the Monitor host and the cluster name. For example, to remove the Monitor running on **host1** from a cluster called **remote**:

   ```
   [root@mon ~]# ceph mon remove host1 --cluster remote
   ```

3. Troubleshoot and fix any problems related to the underlying file system or hardware of the Monitor host.

4. From the Ansible administration node, redeploy the Monitor by running the **ceph-ansible** playbook:

   ```
   $ /usr/share/ceph-ansible/ansible-playbook site.yml
   ```

Additional Resources

- See the *Ceph Monitor is out of quorum* for details.
- The *Managing the storage cluster size* chapter in the *Red Hat Ceph Storage Operations Guide*.
- The *Deploying Red Hat Ceph Storage* chapter in the *Red Hat Ceph Storage 4 Installation Guide*.

4.5. COMPACTING THE MONITOR STORE

When the Monitor store has grown big in size, you can compact it:

- Dynamically by using the **ceph tell** command.
- Upon the start of the **ceph-mon** daemon.
- By using the **ceph-monstore-tool** when the **ceph-mon** daemon is not running. Use this method when the previously mentioned methods fail to compact the Monitor store or when the Monitor is out of quorum and its log contains the **Caught signal (Bus error)** error message.

IMPORTANT

Monitor store size changes when the cluster is not in the **active+clean** state or during the rebalancing process. For this reason, compact the Monitor store when rebalancing is completed. Also, ensure that the placement groups are in the **active+clean** state.

Prerequisites

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

Procedure

1. To compact the Monitor store when the **ceph-mon** daemon is running:

```
ceph tell mon.HOST_NAME compact
```

2. Replace ***HOST_NAME*** with the short host name of the host where the **ceph-mon** is running. Use the **hostname -s** command when unsure.

```
# ceph tell mon.host1 compact
```

3. Add the following parameter to the Ceph configuration under the **[mon]** section:

```
[mon]
mon_compact_on_start = true
```

4. Restart the **ceph-mon** daemon:

```
[root@mon ~]# systemctl restart ceph-mon@_HOST_NAME_
```

Replace ***HOST_NAME*** with the short name of the host where the daemon is running. Use the **hostname -s** command when unsure.

```
[root@mon ~]# systemctl restart ceph-mon@host1
```

5. Ensure that Monitors have formed a quorum:

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

6. Repeat these steps on other Monitors if needed.

NOTE

Before you start, ensure that you have the **ceph-test** package installed.

7. Verify that the **ceph-mon** daemon with the large store is not running. Stop the daemon if needed.

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

Replace ***HOST_NAME*** with the short name of the host where the daemon is running. Use the **hostname -s** command when unsure.

```
[root@mon ~]# systemctl status ceph-mon@host1
[root@mon ~]# systemctl stop ceph-mon@host1
```

8. Compact the Monitor store as a **ceph** user in two different ways:

- Run the command as a **ceph** user:

Syntax

```
su - ceph -c 'ceph-monstore-tool /var/lib/ceph/mon/mon.HOST_NAME compact'
```

Example

```
[root@mon ~]# su - ceph -c 'ceph-monstore-tool /var/lib/ceph/mon/mon.node1 compact'
```

- Run the command as a **root** user and then run **chown** to change the permissions:

a. Run the command as a root user:

Syntax

```
ceph-monstore-tool /var/lib/ceph/mon/mon.HOST_NAME compact
```

Example

```
[root@mon ~]# ceph-monstore-tool /var/lib/ceph/mon/mon.node1 compact
```

b. Change the file permissions:

Example

```
[root@mon ~]# chown -R ceph:ceph /var/lib/ceph/mon/mon.node1
```

9. Start **ceph-mon** again:

```
[root@mon ~]# systemctl start ceph-mon@HOST_NAME
```

Example

```
[root@mon ~]# systemctl start ceph-mon@host1
```

Additional Resources

- *The Ceph Monitor store is getting too big*
- *Ceph Monitor is out of quorum*

4.6. OPENING PORT FOR CEPH MANAGER

The **ceph-mgr** daemons receive placement group information from OSDs on the same range of ports as the **ceph-osd** daemons. If these ports are not open, a cluster will devolve from **HEALTH_OK** to **HEALTH_WARN** and will indicate that PGs are **unknown** with a percentage count of the PGs unknown.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to Ceph manager.

Procedure

1. To resolve this situation, for each host running **ceph-mgr** daemons, open ports **6800-7300**.

   Example

```
[root@ceph-mgr] # firewall-cmd --add-port 6800-7300/tcp
[root@ceph-mgr] # firewall-cmd --add-port 6800-7300/tcp --permanent
```

2. Restart the **ceph-mgr** daemons.

4.7. RECOVERING THE CEPH MONITOR FOR BARE-METAL DEPLOYMENTS

If all the monitors are down in your Red Hat Ceph Storage cluster, and the **ceph -s** command does not execute as expected, you can recover the monitors using the **monmaptool** command. The **monmaptool** command rebuilds the Ceph monitor store from the keyring files of the daemons.

NOTE

This procedure is for **bare-metal** Red Hat Ceph Storage deployments only. For **containerized** Red Hat Ceph Storage deployments, see the Knowledgebase article *MON recovery procedure for Red Hat Ceph Storage containerized deployment when all the three mon are down.*.

Prerequisites

- Bare-metal deployed Red Hat Ceph Storage cluster.
- Root-level access to all the nodes.
- All the Ceph monitors are down.

Procedure

1. Log into the monitor node.
2. From the monitor nodes, if you are unable to access the OSD nodes without being the root user, then copy the public key pair to the OSD nodes:
   a. Generate the SSH key pair, accept the default file name, and leave the passphrase empty:

      Example

      ```
      [root@mons-1 ~]# ssh-keygen
      ```

   b. Copy the public key to **all** the OSD nodes in the storage cluster:

      Example

      ```
      [root@mons-1 ~]#  ssh-copy-id root@osds-1
      [root@mons-1 ~]#  ssh-copy-id root@osds-2
      [root@mons-1 ~]#  ssh-copy-id root@osds-3
      ```

3. Stop the OSD daemon service on all the OSD nodes:

   Example

   ```
   [root@osds-1 ~]#  sudo systemctl stop ceph-osd\*.service ceph-osd.target
   ```

4. To collect the cluster map from all the OSD nodes, create the recovery file and execute the script:
   a. Create the recovery file:

      Example

      ```
      [root@mons-1 ~]# touch recover.sh
      ```

   b. Add the following content to the file and replace the *OSD_NODES* with either the IP addresses of all the OSD nodes or the hostname of the all OSD nodes in the Red Hat Ceph Storage cluster:

      Syntax

      ```
      ------------------------------------------------------------------------ NOTE: The directory
      names specified by 'ms', 'db', and 'db_slow' must end
       with a trailing / otherwise rsync will not operate properly. ------------------------------------
      ----------------------------------
      ```

```
ms=/tmp/monstore/
db=/root/db/
db_slow=/root/db.slow/

mkdir -p $ms $db $db_slow

 ---------------------------------------------------------------------- NOTE: Replace the
contents inside double quotes for 'osd_nodes' below with
 the list of OSD nodes in the environment. ----------------------------------------------------------
------------------
osd_nodes="OSD_NODES_1 OSD_NODES_2 OSD_NODES_3..."

for osd_node in $osd_nodes; do
echo "Operating on $osd_node"
rsync -avz --delete $ms $osd_node:$ms
rsync -avz --delete $db $osd_node:$db
rsync -avz --delete $db_slow $osd_node:$db_slow

ssh -t $osd_node <<EOF
for osd in /var/lib/ceph/osd/ceph-*; do
   ceph-objectstore-tool --type bluestore --data-path \$osd --op update-mon-db --no-
mon-config --mon-store-path $ms
   if [ -e \$osd/keyring ]; then
      cat \$osd/keyring >> $ms/keyring
      echo '    caps mgr = "allow profile osd"' >> $ms/keyring
      echo '    caps mon = "allow profile osd"' >> $ms/keyring
      echo '    caps osd = "allow *"' >> $ms/keyring
   else
      echo WARNING: \$osd on $osd_node does not have a local keyring.
   fi
done
EOF

rsync -avz --delete --remove-source-files $osd_node:$ms $ms
rsync -avz --delete --remove-source-files $osd_node:$db $db
rsync -avz --delete --remove-source-files $osd_node:$db_slow $db_slow
done
 ---------------------------------------------------------------------- End of script
## ----------------------------------------------------------------------
```

c. Provide executable permission on the file:

Example

```
[root@mons-1 ~]# chmod 755 recover.sh
```

d. Execute the file to gather the keyrings of all the OSDs from all the OSD nodes in the storage cluster:

Example

```
[root@mons-1 ~]# ./recovery.sh
```

5. Fetch the keyrings of other daemons from the respective nodes:

a. For Ceph Monitor, the keyring is the same for all the Ceph monitors.

Syntax

```
cat /var/lib/ceph/mon/ceph-MONITOR_NODE/keyring
```

Example

```
[root@mons-1 ~]# cat /var/lib/ceph/mon/ceph-mons-1/keyring
```

b. For Ceph Manager, fetch the keyring from all the manager nodes:

Syntax

```
cat /var/lib/ceph/mgr/ceph-MANAGER_NODE/keyring
```

Example

```
[root@mons-1 ~]# cat /var/lib/ceph/mgr/ceph-mons-1/keyring
```

c. For Ceph OSDs, the keyrings are generated from the above script and stored in a temporary path:
In this example, the OSD keyrings are stored in the **/tmp/monstore/keyring** file.

d. For the client, fetch the keyring from all the client nodes:

Syntax

```
cat /etc/ceph/CLIENT_KEYRING
```

Example

```
[root@client ~]# cat /etc/ceph/ceph.client.admin.keyring
```

e. For meta data server (MDS), fetch the keyring from all the Ceph MDS nodes:

Syntax

```
cat /var/lib/ceph/mds/ceph-MDS_NODE/keyring
```

Example

```
[root@mons-2 ~]# cat /var/lib/ceph/mds/ceph-mds-1/keyring
```

For this key ring append the following caps if not exist:

```
caps mds =  "allow"
caps mon = "allow profile mds"
caps osd = "allow *"
```

f. For Ceph Object Gateway, fetch the keyring from all the Ceph Object Gateway nodes:

Syntax

```
cat /var/lib/ceph/radosgw/ceph-CEPH_OBJECT_GATEWAY_NODE/keyring
```

Example

```
[root@mons-3 ~]# cat /var/lib/ceph/radosgw/ceph-rgw-1/keyring
```

For this key ring append the following caps if not exist:

```
caps mon = "allow rw"
caps osd = "allow *"
```

6. On the Ansible administration node, create a file with all the keyrings fetched from the previous step:

Example

```
[root@admin ~]# cat /tmp/monstore/keyring

[mon.]
 key = AQAa+RxhAAAAABAApmwud0GQHX0raMBc9zTAYQ==
 caps mon = "allow *"
[client.admin]
 key = AQAo+RxhcYWtGBAAiY4kKltMGnAXqPLM2A+B8w==
 caps mds = "allow *"
 caps mgr = "allow *"
 caps mon = "allow *"
 caps osd = "allow *"
[mgr.mons-1]
 key = AQA++RxhAAAAABAAKdG1ETTEMR8KPa9ZpfcIzw==
 caps mds = "allow *"
 caps mon = "allow profile mgr"
 caps osd = "allow *"
[mgr.mons-2]
 key = AQA9+RxhAAAAABAAcCBxsoaIl0sdHTz3dqX4SQ==
 caps mds = "allow *"
 caps mon = "allow profile mgr"
 caps osd = "allow *"
[mgr.mons-3]
 key = AQA/+RxhAAAAABAAwe/mwv0hS79fWP+00W6ypQ==
 caps mds = "allow *"
 caps mon = "allow profile mgr"
 caps osd = "allow *"
[osd.1]
key = AQB/+RxhlH8rFxAAL3mb8Kdb+QuWWdJi+RvwGw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.5]
key = AQCE+RxhKSNsHRAAIyLO5g75tqFVsl6MEEzwXw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
```

```
[osd.8]
key = AQCJ+Rxhc0wHJhAA5Bb2kU9Nadpm3UCLASnCfw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.2]
key = AQB/+RxhhrQCGRAAUhh77gIVhN8zsTbaKMJuHw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.4]
key = AQCE+Rxh0mDxDRAApAeqKOJycW5bpP3IuAhSMw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.7]
key = AQCJ+Rxhn+RAIhAAp1ImK1jiazBsDpmTQvVEVw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.0]
key = AQB/+RxhPhh+FRAAc5b0nwiuK6o1AIbjVc6tQg==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.3]
key = AQCE+RxhJv8PARAAqCzH2br1xJmMTNnqH3I9mA==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.6]
key = AQCI+RxhAt4eIhAAYQEJqSNRT7I2WNl/rYQcKQ==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.1]
key = AQB/+RxhlH8rFxAAL3mb8Kdb+QuWWdJi+RvwGw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.5]
key = AQCE+RxhKSNsHRAAIyLO5g75tqFVsl6MEEzwXw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.8]
key = AQCJ+Rxhc0wHJhAA5Bb2kU9Nadpm3UCLASnCfw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.2]
key = AQB/+RxhhrQCGRAAUhh77gIVhN8zsTbaKMJuHw==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.0]
```

```
key = AQB/+RxhPhh+FRAAc5b0nwiuK6o1AIbjVc6tQg==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.3]
key = AQCE+RxhJv8PARAAqCzH2br1xJmMTNnqH3I9mA==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[osd.6]
key = AQCI+RxhAt4eIhAAYQEJqSNRT7I2WNl/rYQcKQ==
    caps mgr = "allow profile osd"
    caps mon = "allow profile osd"
    caps osd = "allow *"
[mds.mds-1]
 key = AQDs+RxhAF9vERAAdn6ArdUJ31RLr2sBVkzp3A==
        caps mds = "allow"
        caps mon = "allow profile mds"
        caps osd = "allow *"
[mds.mds-2]
 key = AQDs+RxhROoAFxAALAgMfM45wC5ht/vSFN2EzQ==
        caps mds = "allow"
        caps mon = "allow profile mds"
        caps osd = "allow *"
[mds.mds-3]
 key = AQDs+Rxhd092FRAArXLIHAhMp2z9zcWDCSoIDQ==
        caps mds = "allow"
        caps mon = "allow profile mds"
        caps osd = "allow *"
[client.rgw.rgws-1.rgw0]
 key = AQD9+Rxh0iP2MxAAYY76Js1AaZhzFG44cvcyOw==
        caps mon = "allow rw"
        caps osd = "allow *"
```

7. Optional: On each Ceph Monitor node, ensure that the monitor map is not available:

Example

```
[root@mons-1 ~]# ceph-monstore-tool /tmp/monstore get monmap -- --out /tmp/monmap
[root@mons-1 ~]# monmaptool /tmp/monmap --print

monmaptool: monmap file /tmp/monmap
monmaptool: couldn't open /tmp/monmap: (2) No such file or directory

Notice theNo such file or directory  error message if monmap is missed

Notice that the “No such file or directory”  error message if monmap is missed
```

8. From each Ceph Monitor node, fetch the *MONITOR_ID*, *IP_ADDRESS_OF_MONITOR*, and *FSID* from the **etc/ceph/ceph.conf** file:

Example

```
[global]
cluster network = 10.0.208.0/22
```

```
fsid = 9877bde8-ccb2-4758-89c3-90ca9550ffea
mon host = [v2:10.0.211.00:3300,v1:10.0.211.00:6789],
[v2:10.0.211.13:3300,v1:10.0.211.13:6789],[v2:10.0.210.13:3300,v1:10.0.210.13:6789]
mon initial members = ceph-mons-1, ceph-mons-2, ceph-mons-3
```

9. On the Ceph Monitor node, rebuild the monitor map:

Syntax

```
monmaptool --create --addv MONITOR_ID IP_ADDRESS_OF_MONITOR --enable-all-features --clobber PATH_OF_MONITOR_MAP --fsid FSID
```

Example

```
[root@mons-1 ~]# monmaptool --create --addv mons-1
[v2:10.74.177.30:3300,v1:10.74.177.30:6789] --addv mons-2
[v2:10.74.179.197:3300,v1:10.74.179.197:6789] --addv mons-3
[v2:10.74.182.123:3300,v1:10.74.182.123:6789] --enable-all-features --clobber
/root/monmap.mons-1 --fsid 6c01cb34-33bf-44d0-9aec-3432276f6be8

monmaptool: monmap file /root/monmap.mons-1
monmaptool: set fsid to 6c01cb34-33bf-44d0-9aec-3432276f6be8
monmaptool: writing epoch 0 to /root/monmap.mon-a (3 monitors)
```

10. On the Ceph Monitor node, check the generated monitor map:

Syntax

```
monmaptool PATH_OF_MONITOR_MAP --print
```

Example

```
[root@mons-1 ~]# monmaptool /root/monmap.mons-1 --print

monmaptool: monmap file /root/monmap.mons-1
epoch 0
fsid 6c01cb34-33bf-44d0-9aec-3432276f6be8
last_changed 2021-11-23 02:57:23.235505
created 2021-11-23 02:57:23.235505
min_mon_release 0 (unknown)
election_strategy: 1
0: [v2:10.74.177.30:3300/0,v1:10.74.177.30:6789/0] mon.mons-1
1: [v2:10.74.179.197:3300/0,v1:10.74.179.197:6789/0] mon.mons-2
2: [v2:10.74.182.123:3300/0,v1:10.74.182.123:6789/0] mon.mons-3
```

11. On the Ceph Monitor node where we are recovering the monitors, rebuild the Ceph Monitor store from the collected map:

Syntax

```
ceph-monstore-tool /tmp/monstore rebuild -- --keyring KEYRING_PATH --monmap PATH_OF_MONITOR_MAP
```

In this example, the recovery is run on the **mons-1** node.

Example

```
[root@mons-1 ~]# ceph-monstore-tool /tmp/monstore rebuild -- --keyring
/tmp/monstore/keyring  --monmap /root/monmap.mons-1
```

12. Change the ownership of monstore directory to ceph:

Example

```
[root@mons-1 ~]# chown -R ceph:ceph /tmp/monstore
```

13. On all the Ceph Monitor nodes, take a back-up of the corrupted store:

Example

```
[root@mons-1 ~]# mv /var/lib/ceph/mon/ceph-mons-1/store.db /var/lib/ceph/mon/ceph-mons-
1/store.db.corrupted
```

14. On all the Ceph Monitor nodes, replace the corrupted store:

Example

```
[root@mons-1 ~]# scp -r /tmp/monstore/store.db mons-1:/var/lib/ceph/mon/ceph-mons-1/
```

15. On all the Ceph Monitor nodes, change the owner of the new store:

Example

```
[root@mons-1 ~]# chown -R ceph:ceph /var/lib/ceph/mon/ceph-HOSTNAME/store.db
```

16. On all the Ceph OSD nodes, start the OSDs:

Example

```
[root@osds-1 ~]# sudo systemctl start ceph-osd.target
```

17. On all the Ceph Monitor nodes, start the monitors

Example

```
[root@mons-1 ~]# sudo systemctl start ceph-mon.target
```

4.8. RECOVERING THE CEPH MONITOR STORE

Ceph Monitors store the cluster map in a key–value store such as LevelDB. If the store is corrupted on a Monitor, the Monitor terminates unexpectedly and fails to start again. The Ceph logs might include the following errors:

```
Corruption: error in middle of record
Corruption: 1 missing files; e.g.: /var/lib/ceph/mon/mon.0/store.db/1234567.ldb
```

Production Red Hat Ceph Storage clusters use at least three Ceph Monitors so that if one fails, it can be replaced with another one. However, under certain circumstances, all Ceph Monitors can have corrupted stores. For example, when the Ceph Monitor nodes have incorrectly configured disk or file system settings, a power outage can corrupt the underlying file system.

If there is corruption on all Ceph Monitors, you can recover it with information stored on the OSD nodes by using utilities called **ceph-monstore-tool** and **ceph-objectstore-tool**.

IMPORTANT

These procedures cannot recover the following information:

- Metadata Daemon Server (MDS) keyrings and maps
- Placement Group settings:
  - **full ratio** set by using the **ceph pg set_full_ratio** command
  - **nearfull ratio** set by using the **ceph pg set_nearfull_ratio** command

IMPORTANT

Never restore the Ceph Monitor store from an old backup. Rebuild the Ceph Monitor store from the current cluster state using the following steps and restore from that.

4.8.1. Recovering the Ceph Monitor store when using BlueStore

Follow this procedure if the Ceph Monitor store is corrupted on all Ceph Monitors and you use the BlueStore back end.

In containerized environments, this method requires attaching Ceph repositories and restoring to a non-containerized Ceph Monitor first.

WARNING

This procedure can cause data loss. If you are unsure about any step in this procedure, contact the Red Hat Technical Support for an assistance with the recovering process.

Prerequisites

- **Bare-metal deployments**
  - The **rsync** and **ceph-test** packages are installed.
- **Container deployments**
  - All OSDs containers are stopped.

- Enable Ceph repositories on the Ceph nodes based on their roles.
- The **ceph-test** and **rsync** packages are installed on the OSD and Monitor nodes.
- The **ceph-mon** package is installed on the Monitor nodes.
- The **ceph-osd** package is installed on the OSD nodes.

Procedure

1. If you use Ceph in **containers**, mount all disk with Ceph data to a temporary location. Repeat this step for all OSD nodes.

   a. List the data partitions. Use **ceph-volume** or **ceph-disk** depending on which utility you used to set up the devices:

   ```
   [root@osd ~]# ceph-volume lvm list
   ```

   or

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

   b. Mount the data partitions to temporary location:

   ```
   mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-$i
   ```

   c. Restore the SELinux context:

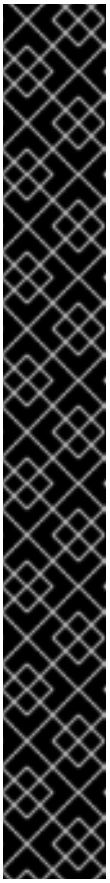
   ```
   for i in {OSD_ID}; do restorecon /var/lib/ceph/osd/ceph-$i; done
   ```

   Replace *OSD_ID* with a numeric, space-separated list of Ceph OSD IDs on the OSD node.

   d. Change the owner and group to **ceph:ceph**:

   ```
   for i in {OSD_ID}; do chown -R ceph:ceph /var/lib/ceph/osd/ceph-$i; done
   ```

   Replace *OSD_ID* with a numeric, space-separated list of Ceph OSD IDs on the OSD node.



IMPORTANT

Due to a bug that causes the **update-mon-db** command to use additional **db** and **db.slow** directories for the Monitor database, you must also copy these directories. To do so:

1. Prepare a temporary location outside the container to mount and access the OSD database and extract the OSD maps needed to restore the Ceph Monitor:

   ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev *OSD-DATA* --path /var/lib/ceph/osd/ceph-*OSD-ID*

   Replace *OSD-DATA* with the Volume Group (VG) or Logical Volume (LV) path to the to the OSD data and *OSD-ID* with the ID of the OSD.

2. Create a symbolic link between the BlueStore database and **block.db**:

   ln -snf *BLUESTORE DATABASE* /var/lib/ceph/osd/ceph-*OSD-ID*/block.db

   Replace *BLUESTORE-DATABASE* with the Volume Group (VG) or Logical Volume (LV) path to the BlueStore database and *OSD-ID* with the ID of the OSD.

2. Use the following commands from the Ceph Monitor node with the corrupted store. Repeat them for all OSDs on all nodes.

   a. Collect the cluster map from all OSD nodes:

```
[root@ mon~]# cd /root/
[root@mon ~]# ms=/tmp/monstore/
[root@mon ~]# db=/root/db/
[root@mon ~]# db_slow=/root/db.slow/

[root@mon ~]# mkdir $ms
[root@mon ~]# for host in $osd_nodes; do
              echo "$host"
              rsync -avz $ms $host:$ms
              rsync -avz $db $host:$db
              rsync -avz $db_slow $host:$db_slow

              rm -rf $ms
              rm -rf $db
              rm -rf $db_slow

              sh -t $host <<EOF
                for osd in /var/lib/ceph/osd/ceph-*; do
                  ceph-objectstore-tool --type bluestore --data-path \$osd --op update-mon-db
--mon-store-path $ms

                done
              EOF

                  rsync -avz $host:$ms $ms
```

```
        rsync -avz $host:$db $db
        rsync -avz $host:$db_slow $db_slow
    done
```

b. Set the appropriate capabilities:

```
[root@mon ~]# ceph-authtool /etc/ceph/ceph.client.admin.keyring -n mon. --cap mon
'allow *' --gen-key
[root@mon ~]# cat /etc/ceph/ceph.client.admin.keyring
  [mon.]
    key = AQCleqldWqm5IhAAgZQbEzoShkZV42RiQVffnA==
    caps mon = "allow *"
  [client.admin]
    key = AQCmAKld8J05KxAArOWeRAw63gAwwZO5o75ZNQ==
    auid = 0
    caps mds = "allow *"
    caps mgr = "allow *"
    caps mon = "allow *"
    caps osd = "allow *"
```

c. Move all **sst** file from the **db** and **db.slow** directories to the temporary location:

```
[root@mon ~]# mv /root/db/*.sst /root/db.slow/*.sst /tmp/monstore/store.db
```

d. Rebuild the Monitor store from the collected map:

```
[root@mon ~]# ceph-monstore-tool /tmp/monstore rebuild -- --keyring
/etc/ceph/ceph.client.admin
```

NOTE

After using this command, only keyrings extracted from the OSDs and the keyring specified on the **ceph-monstore-tool** command line are present in Ceph's authentication database. You have to recreate or import all other keyrings, such as clients, Ceph Manager, Ceph Object Gateway, and others, so those clients can access the cluster.

e. Back up the corrupted store. Repeat this step for all Ceph Monitor nodes:

```
mv /var/lib/ceph/mon/ceph-HOSTNAME/store.db
/var/lib/ceph/mon/ceph-HOSTNAME/store.db.corrupted
```

Replace *HOSTNAME* with the host name of the Ceph Monitor node.

f. Replace the corrupted store. Repeat this step for all Ceph Monitor nodes:

```
scp -r /tmp/monstore/store.db HOSTNAME:/var/lib/ceph/mon/ceph-HOSTNAME/
```

Replace *HOSTNAME* with the host name of the Monitor node.

g. Change the owner of the new store. Repeat this step for all Ceph Monitor nodes:

```
chown -R ceph:ceph /var/lib/ceph/mon/ceph-HOSTNAME/store.db
```

Replace *HOSTNAME* with the host name of the Ceph Monitor node.

3. If you use Ceph **in containers**, then unmount all the temporary mounted OSDs on all nodes:

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

4. Start all the Ceph Monitor daemons:

```
[root@mon ~]# systemctl start ceph-mon *
```

5. Ensure that the Monitors are able to form a quorum:

   - **Bare-metal deployments**

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

   - **Containers**

     ```
     [user@admin ~]$ docker exec ceph-mon-_HOSTNAME_ ceph -s
     ```

     Replace *HOSTNAME* with the host name of the Ceph Monitor node.

6. Import the Ceph Manager keyring and start all Ceph Manager processes:

```
ceph auth import -i /etc/ceph/ceph.mgr.HOSTNAME.keyring
systemctl start ceph-mgr@HOSTNAME
```

   Replace *HOSTNAME* with the host name of the Ceph Manager node.

7. Start all OSD processes across all OSD nodes:

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

8. Ensure that the OSDs are returning to service:

   - **Bare-metal deployments**

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

   - **Containers**

     ```
     [user@admin ~]$ docker exec ceph-mon-_HOSTNAME_ ceph -s
     ```

     Replace *HOSTNAME* with the host name of the Ceph Monitor node.

Additional Resources

- For details on registering Ceph nodes to the Content Delivery Network (CDN), see *Registering Red Hat Ceph Storage Nodes to the CDN and Attaching Subscriptions* section in the *Red Hat Ceph Storage Installation Guide*.
- For details on enabling repositories, see the *Enabling the Red Hat Ceph Storage Repositories* section in the *Red Hat Ceph Storage Installation Guide*.

4.9. ADDITIONAL RESOURCES

- See Chapter 3, *Troubleshooting networking issues* in the *Red Hat Ceph Storage Troubleshooting Guide* for network-related problems.

CHAPTER 5. TROUBLESHOOTING CEPH OSDS

This chapter contains information on how to fix the most common errors related to Ceph OSDs.

5.1. PREREQUISITES

- Verify your network connection. See *Troubleshooting networking issues* for details.
- Verify that Monitors have a quorum by using the **ceph health** command. If the command returns a health status (**HEALTH_OK**, **HEALTH_WARN**, or **HEALTH_ERR**), the Monitors are able to form a quorum. If not, address any Monitor problems first. See *Troubleshooting Ceph Monitors* for details. For details about **ceph health** see *Understanding Ceph health*.
- Optionally, stop the rebalancing process to save time and resources. See *Stopping and starting rebalancing* for details.

5.2. MOST COMMON CEPH OSD ERRORS

The following tables list the most common error messages that are returned by the **ceph health detail** command, or included in the Ceph logs. The tables provide links to corresponding sections that explain the errors and point to specific procedures to fix the problems.

5.2.1. Prerequisites

- Root-level access to the Ceph OSD nodes.

5.2.2. Ceph OSD error messages

A table of common Ceph OSD error messages, and a potential fix.

Error message	See
HEALTH_ERR	
full osds	Full OSDS
HEALTH_WARN	
backfillfull osds	Backfillfull OSDS
nearfull osds	Nearfull OSDS
osds are down	OSDS are down Flapping OSDS
requests are blocked	Slow request or requests are blocked
slow requests	Slow request or requests are blocked

5.2.3. Common Ceph OSD error messages in the Ceph logs

A table of common Ceph OSD error messages found in the Ceph logs, and a link to a potential fix.

Error message	Log file	See
heartbeat_check: no reply from osd.X	Main cluster log	Flapping OSDS
wrongly marked me down	Main cluster log	Flapping OSDS
osds have slow requests	Main cluster log	Slow request or requests are blocked
FAILED assert(0 == "hit suicide timeout")	OSD log	Down OSDs

5.2.4. Full OSDs

The **ceph health detail** command returns an error message similar to the following one:

```
HEALTH_ERR 1 full osds
osd.3 is full at 95%
```

What This Means

Ceph prevents clients from performing I/O operations on full OSD nodes to avoid losing data. It returns the **HEALTH_ERR full osds** message when the cluster reaches the capacity set by the **mon_osd_full_ratio** parameter. By default, this parameter is set to **0.95** which means 95% of the cluster capacity.

To Troubleshoot This Problem

Determine how many percent of raw storage (**%RAW USED**) is used:

```
# ceph df
```

If **%RAW USED** is above 70-75%, you can:

- Delete unnecessary data. This is a short-term solution to avoid production downtime.
- Scale the cluster by adding a new OSD node. This is a long-term solution recommended by Red Hat.

Additional Resources

- *Nearfull OSDS* in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See *Deleting data from a full storage cluster* for details.

5.2.5. Backfillfull OSDs

The **ceph health detail** command returns an error message similar to the following one:

```
health: HEALTH_WARN
3 backfillfull osd(s)
Low space hindering backfill (add storage if this doesn't resolve itself): 32 pgs backfill_toofull
```

What this means

When one or more OSDs has exceeded the backfillfull threshold, Ceph prevents data from rebalancing to this device. This is an early warning that rebalancing might not complete and that the cluster is approaching full. The default for the backfullfull threshold is 90%.

To troubleshoot this problem

Check utilization by pool:

```
ceph df
```

If **%RAW USED** is above 70-75%, you can carry out one of the following actions:

- Delete unnecessary data. This is a short-term solution to avoid production downtime.
- Scale the cluster by adding a new OSD node. This is a long-term solution recommended by Red Hat.
- Increase the **backfillfull** ratio for the OSDs that contain the PGs stuck in **backfull_toofull** to allow the recovery process to continue. Add new storage to the cluster as soon as possible or remove data to prevent filling more OSDs.

  #### Syntax

  ```
  ceph osd set-backfillfull-ratio VALUE
  ```

  The range for *VALUE* is 0.0 to 1.0.

  #### Example

  ```
  [ceph: root@host01/]# ceph osd set-backfillfull-ratio 0.92
  ```

Additional Resources

- *Nearfull OSDS* in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See *Deleting data from a full storage cluster* for details.

5.2.6. Nearfull OSDs

The **ceph health detail** command returns an error message similar to the following one:

```
HEALTH_WARN 1 nearfull osds
osd.2 is near full at 85%
```

What This Means

Ceph returns the **nearfull osds** message when the cluster reaches the capacity set by the **mon osd nearfull ratio defaults** parameter. By default, this parameter is set to **0.85** which means 85% of the cluster capacity.

Ceph distributes data based on the CRUSH hierarchy in the best possible way but it cannot guarantee equal distribution. The main causes of the **uneven data distribution** and the **nearfull osds** messages are:

- The OSDs are not balanced among the OSD nodes in the cluster. That is, some OSD nodes host significantly more OSDs than others, or the weight of some OSDs in the CRUSH map is not adequate to their capacity.
- The Placement Group (PG) count is not proper as per the number of the OSDs, use case, target PGs per OSD, and OSD utilization.
- The cluster uses inappropriate CRUSH tunables.
- The back-end storage for OSDs is almost full.

To Troubleshoot This Problem:

1. Verify that the PG count is sufficient and increase it if needed.
2. Verify that you use CRUSH tunables optimal to the cluster version and adjust them if not.
3. Change the weight of OSDs by utilization.
4. Enable the Ceph Manager balancer module which optimizes the placement of placement groups (PGs) across OSDs in order to achieve a balanced distribution

   Example

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

5. Determine how much space is left on the disks used by OSDs.
   a. To view how much space OSDs use in general:

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

   b. To view how much space OSDs use on particular nodes. Use the following command from the node containing **nearful** OSDs:

      ```
      $ df
      ```

   c. If needed, add a new OSD node.

Additional Resources

- *Full OSDs*
- See the *Using the Ceph Manager balancer module* section in the *Red Hat Ceph Storage Operations Guide*.

- See the *Set an OSD's Weight by Utilization* section in the *Storage Strategies* guide for Red Hat Ceph Storage 4.
- For details, see the *CRUSH Tunables* section in the *Storage Strategies* guide for Red Hat Ceph Storage 4 and the *How can I test the impact CRUSH map tunable modifications will have on my PG distribution across OSDs in Red Hat Ceph Storage?* solution on the Red Hat Customer Portal.
- See *Increasing the placement group* for details.

5.2.7. Down OSDs

The **ceph health** command returns an error similar to the following one:

```
HEALTH_WARN 1/3 in osds are down
```

What This Means

One of the **ceph-osd** processes is unavailable due to a possible service failure or problems with communication with other OSDs. As a consequence, the surviving **ceph-osd** daemons reported this failure to the Monitors.

If the **ceph-osd** daemon is not running, the underlying OSD drive or file system is either corrupted, or some other error, such as a missing keyring, is preventing the daemon from starting.

In most cases, networking issues cause the situation when the **ceph-osd** daemon is running but still marked as **down**.

To Troubleshoot This Problem

1. Determine which OSD is **down**:

```
[root@mon ~]# ceph health detail
HEALTH_WARN 1/3 in osds are down
osd.0 is down since epoch 23, last address 192.168.106.220:6800/11080
```

2. Try to restart the **ceph-osd** daemon:

```
[root@mon ~]# systemctl restart ceph-osd@OSD_NUMBER
```

Replace ***OSD_NUMBER*** with the ID of the OSD that is **down**, for example:

```
[root@mon ~]# systemctl restart ceph-osd@0
```

   a. If you are not able start **ceph-osd**, follow the steps in *The **ceph-osd** daemon cannot start*.

   b. If you are able to start the **ceph-osd** daemon but it is marked as **down**, follow the steps in *The **ceph-osd** daemon is running but still marked as `down`*.

The ceph-osd daemon cannot start

1. If you have a node containing a number of OSDs (generally, more than twelve), verify that the default maximum number of threads (PID count) is sufficient. See *Increasing the PID count* for details.

2. Verify that the OSD data and journal partitions are mounted properly. You can use the **ceph-volume lvm list** command to list all devices and volumes associated with the Ceph Storage Cluster and then manually inspect if they are mounted properly. See the **mount(8)** manual page for details.

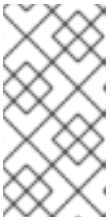
3. If you got the **ERROR: missing keyring, cannot use cephx for authentication** error message, the OSD is a missing keyring.

4. If you got the **ERROR: unable to open OSD superblock on /var/lib/ceph/osd/ceph-1** error message, the **ceph-osd** daemon cannot read the underlying file system. See the following steps for instructions on how to troubleshoot and fix this error.

NOTE

If this error message is returned during boot time of the OSD host, open a support ticket as this might indicate a known issue tracked in the Red Hat Bugzilla 1439210.

5. Check the corresponding log file to determine the cause of the failure. By default, Ceph stores log files in the **/var/log/ceph/** directory for bare-metal deployments.



NOTE

For container-based deployment, Ceph generates logs to **journald**. You can enable logging to files in **/var/log/ceph** by setting **log_to_file** parameter to **true** under [global] in the Ceph configuration file. See *Understanding ceph logs* for more details.

   a. An **EIO** error message similar to the following one indicates a failure of the underlying disk: To fix this problem replace the underlying OSD disk. See *Replacing an OSD drive* for details.

   b. If the log includes any other **FAILED assert** errors, such as the following one, open a support ticket. See *Contacting Red Hat Support for service* for details.

```
FAILED assert(0 == "hit suicide timeout")
```

6. Check the **dmesg** output for the errors with the underlying file system or disk:

```
$ dmesg
```

   a. If the **dmesg** output includes any **SCSI error** error messages, see the *SCSI Error Codes Solution Finder* solution on the Red Hat Customer Portal to determine the best way to fix the problem.

   b. Alternatively, if you are unable to fix the underlying file system, replace the OSD drive. See *Replacing an OSD drive* for details.

7. If the OSD failed with a segmentation fault, such as the following one, gather the required information and open a support ticket. See *Contacting Red Hat Support for service* for details.

```
Caught signal (Segmentation fault)
```

The ceph-osd is running but still marked as down

1. Check the corresponding log file to determine the cause of the failure. By default, Ceph stores log files in the **/var/log/ceph/** directory in bare-metal deployments.

NOTE

For container-based deployment, Ceph generates logs to **journald**. You can enable logging to files in **/var/log/ceph** by setting **log_to_file** parameter to **true** under [global] in the Ceph configuration file. See *Understanding ceph logs* for more details.

   a. If the log includes error messages similar to the following ones, see *Flapping OSDs*.

```
wrongly marked me down
heartbeat_check: no reply from osd.2 since back
```

   b. If you see any other errors, open a support ticket. See *Contacting Red Hat Support for service* for details.

Additional Resources

- *Flapping OSDs*
- *Stale placement groups*
- See the *Starting, stopping, restarting the Ceph daemon by instances* section in the *Red Hat Ceph Storage Administration Guide*.
- See the *Managing Ceph keyrings* section in the *Red Hat Ceph Storage Administration Guide*.

5.2.8. Flapping OSDs

The **ceph -w | grep osds** command shows OSDs repeatedly as **down** and then **up** again within a short period of time:

```
# ceph -w | grep osds
2021-04-05 06:27:20.810535 mon.0 [INF] osdmap e609: 9 osds: 8 up, 9 in
2021-04-05 06:27:24.120611 mon.0 [INF] osdmap e611: 9 osds: 7 up, 9 in
2021-04-05 06:27:25.975622 mon.0 [INF] HEALTH_WARN; 118 pgs stale; 2/9 in osds are down
2021-04-05 06:27:27.489790 mon.0 [INF] osdmap e614: 9 osds: 6 up, 9 in
2021-04-05 06:27:36.540000 mon.0 [INF] osdmap e616: 9 osds: 7 up, 9 in
2021-04-05 06:27:39.681913 mon.0 [INF] osdmap e618: 9 osds: 8 up, 9 in
2021-04-05 06:27:43.269401 mon.0 [INF] osdmap e620: 9 osds: 9 up, 9 in
2021-04-05 06:27:54.884426 mon.0 [INF] osdmap e622: 9 osds: 8 up, 9 in
2021-04-05 06:27:57.398706 mon.0 [INF] osdmap e624: 9 osds: 7 up, 9 in
2021-04-05 06:27:59.669841 mon.0 [INF] osdmap e625: 9 osds: 6 up, 9 in
2021-04-05 06:28:07.043677 mon.0 [INF] osdmap e628: 9 osds: 7 up, 9 in
2021-04-05 06:28:10.512331 mon.0 [INF] osdmap e630: 9 osds: 8 up, 9 in
2021-04-05 06:28:12.670923 mon.0 [INF] osdmap e631: 9 osds: 9 up, 9 in
```

In addition the Ceph log contains error messages similar to the following ones:

```
2021-07-25 03:44:06.510583 osd.50 127.0.0.1:6801/149046 18992 : cluster [WRN] map e600547
wrongly marked me down
```

```
2021-07-25 19:00:08.906864 7fa2a0033700 -1 osd.254 609110 heartbeat_check: no reply from
osd.2 since back 2021-07-25 19:00:07.444113 front 2021-07-25 18:59:48.311935 (cutoff 2021-07-25
18:59:48.906862)
```

What This Means

The main causes of flapping OSDs are:

- Certain storage cluster operations, such as scrubbing or recovery, take an abnormal amount of time, for example if you perform these operations on objects with a large index or large placement groups. Usually, after these operations finish, the flapping OSDs problem is solved.
- Problems with the underlying physical hardware. In this case, the **ceph health detail** command also returns the **slow requests** error message.
- Problems with network.

Ceph OSDs cannot manage situations where the private network for the storage cluster fails, or significant latency is on the public client-facing network.

Ceph OSDs use the private network for sending heartbeat packets to each other to indicate that they are **up** and **in**. If the private storage cluster network does not work properly, OSDs are unable to send and receive the heartbeat packets. As a consequence, they report each other as being **down** to the Ceph Monitors, while marking themselves as **up**.

The following parameters in the Ceph configuration file influence this behavior:

Parameter	Description	Default value
osd_heartbeat_grace_time	How long OSDs wait for the heartbeat packets to return before reporting an OSD as **down** to the Ceph Monitors.	20 seconds
mon_osd_min_down_reporters	How many OSDs must report another OSD as **down** before the Ceph Monitors mark the OSD as **down**	2

This table shows that in default configuration, the Ceph Monitors mark an OSD as **down** if only one OSD made three distinct reports about the first OSD being **down**. In some cases, if one single host encounters network issues, the entire cluster can experience flapping OSDs. This is because the OSDs that reside on the host will report other OSDs in the cluster as **down**.

NOTE

The flapping OSDs scenario does not include the situation when the OSD processes are started and then immediately killed.

To Troubleshoot This Problem

1. Check the output of the **ceph health detail** command again. If it includes the **slow requests** error message, see for details on how to troubleshoot this issue.

```
# ceph health detail
HEALTH_WARN 30 requests are blocked > 32 sec; 3 osds have slow requests
30 ops are blocked > 268435 sec
```

```
1 ops are blocked > 268435 sec on osd.11
1 ops are blocked > 268435 sec on osd.18
28 ops are blocked > 268435 sec on osd.39
3 osds have slow requests
```

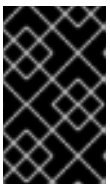
2. Determine which OSDs are marked as **down** and on what nodes they reside:

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

3. On the nodes containing the flapping OSDs, troubleshoot and fix any networking problems. For details, see *Troubleshooting networking issues*.

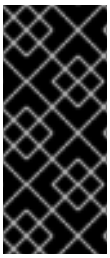
4. Alternatively, you can temporary force Monitors to stop marking the OSDs as **down** and **up** by setting the **noup** and **nodown** flags:

```
# ceph osd set noup
# ceph osd set nodown
```



IMPORTANT

Using the **noup** and **nodown** flags does not fix the root cause of the problem but only prevents OSDs from flapping. To open a support ticket, see the *Contacting Red Hat Support for service* section for details.



IMPORTANT

Flapping OSDs can be caused by MTU misconfiguration on Ceph OSD nodes, at the network switch level, or both. To resolve the issue, set MTU to a uniform size on all storage cluster nodes, including on the core and access network switches with a planned downtime. Do not tune **osd heartbeat min size** because changing this setting can hide issues within the network, and it will not solve actual network inconsistency.

Additional Resources

- See the *Verifying the Network Configuration for Red Hat Ceph Storage* section in the *Red Hat Ceph Storage Installation Guide* for details.
- See the *Ceph heartbeat* section in the *Red Hat Ceph Storage Architecture Guide* for details.
- See the *Slow requests or requests are blocked* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See Red Hat's Knowledgebase solution *How to reduce scrub impact in a Red Hat Ceph Storage cluster?* for tuning scrubbing process.

5.2.9. Slow requests or requests are blocked

The **ceph-osd** daemon is slow to respond to a request and the **ceph health detail** command returns an error message similar to the following one:

```
HEALTH_WARN 30 requests are blocked > 32 sec; 3 osds have slow requests
30 ops are blocked > 268435 sec
1 ops are blocked > 268435 sec on osd.11
```

```
1 ops are blocked > 268435 sec on osd.18
28 ops are blocked > 268435 sec on osd.39
3 osds have slow requests
```

In addition, the Ceph logs include an error message similar to the following ones:

```
2015-08-24 13:18:10.024659 osd.1 127.0.0.1:6812/3032 9 : cluster [WRN] 6 slow requests, 6 included below; oldest blocked for > 61.758455 secs
```

```
2016-07-25 03:44:06.510583 osd.50 [WRN] slow request 30.005692 seconds old, received at {date-time}: osd_op(client.4240.0:8 benchmark_data_ceph-1_39426_object7 [write 0~4194304] 0.69848840) v4 currently waiting for subops from [610]
```

What This Means

An OSD with slow requests is every OSD that is not able to service the I/O operations per second (IOPS) in the queue within the time defined by the **osd_op_complaint_time** parameter. By default, this parameter is set to 30 seconds.

The main causes of OSDs having slow requests are:

- Problems with the underlying hardware, such as disk drives, hosts, racks, or network switches
- Problems with network. These problems are usually connected with flapping OSDs. See *Flapping OSDs* for details.
- System load

The following table shows the types of slow requests. Use the **dump_historic_ops** administration socket command to determine the type of a slow request. For details about the administration socket, see the *Using the Ceph Administration Socket* section in the *Administration Guide* for Red Hat Ceph Storage 4.

Slow request type	Description
waiting for rw locks	The OSD is waiting to acquire a lock on a placement group for the operation.
waiting for subops	The OSD is waiting for replica OSDs to apply the operation to the journal.
no flag points reached	The OSD did not reach any major operation milestone.
waiting for degraded object	The OSDs have not replicated an object the specified number of times yet.

To Troubleshoot This Problem

1. Determine if the OSDs with slow or block requests share a common piece of hardware, for example a disk drive, host, rack, or network switch.
2. If the OSDs share a disk:

a. Use the **smartmontools** utility to check the health of the disk or the logs to determine any errors on the disk.

NOTE

The **smartmontools** utility is included in the **smartmontools** package.

b. Use the **iostat** utility to get the I/O wait report (**%iowai**) on the OSD disk to determine if the disk is under heavy load.

NOTE

The **iostat** utility is included in the **sysstat** package.

3. If the OSDs share the node with another service:

a. Check the RAM and CPU utilization

b. Use the **netstat** utility to see the network statistics on the Network Interface Controllers (NICs) and troubleshoot any networking issues. See also *Troubleshooting networking issues* for further information.

4. If the OSDs share a rack, check the network switch for the rack. For example, if you use jumbo frames, verify that the NIC in the path has jumbo frames set.

5. If you are unable to determine a common piece of hardware shared by OSDs with slow requests, or to troubleshoot and fix hardware and networking problems, open a support ticket. See *Contacting Red Hat support for service* for details.

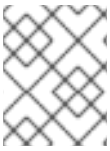
Additional Resources

- See the *Using the Ceph Administration Socket* section in the *Red Hat Ceph Storage Administration Guide* for details.

5.3. STOPPING AND STARTING REBALANCING

When an OSD fails or you stop it, the CRUSH algorithm automatically starts the rebalancing process to redistribute data across the remaining OSDs.

Rebalancing can take time and resources, therefore, consider stopping rebalancing during troubleshooting or maintaining OSDs.



NOTE

Placement groups within the stopped OSDs become **degraded** during troubleshooting and maintenance.

Prerequisites

- Root-level access to the Ceph Monitor node.

Procedure

1. To do so, set the **noout** flag before stopping the OSD:

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

2. When you finish troubleshooting or maintenance, unset the **noout** flag to start rebalancing:

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

Additional Resources

- The *Rebalancing and Recovery* section in the *Red Hat Ceph Storage Architecture Guide*.

5.4. MOUNTING THE OSD DATA PARTITION

If the OSD data partition is not mounted correctly, the **ceph-osd** daemon cannot start. If you discover that the partition is not mounted as expected, follow the steps in this section to mount it.

This section is specific to bare-metal deployments only.

Prerequisites

- Access to the **ceph-osd** daemon.
- Root-level access to the Ceph Monitor node.

Procedure

1. Mount the partition:

```
[root@ceph-mon]# mount -o noatime PARTITION
/var/lib/ceph/osd/CLUSTER_NAME-OSD_NUMBER
```

Replace ***PARTITION*** with the path to the partition on the OSD drive dedicated to OSD data. Specify the cluster name and the OSD number.

Example

```
[root@ceph-mon]# mount -o noatime /dev/sdd1 /var/lib/ceph/osd/ceph-0
```

2. Try to start the failed **ceph-osd** daemon:

```
[root@ceph-mon]# systemctl start ceph-osd@OSD_NUMBER
```

Replace the ***OSD_NUMBER*** with the ID of the OSD.

Example

```
[root@ceph-mon]# systemctl start ceph-osd@0
```

Additional Resources

- See the *Down OSDs* in the *Red Hat Ceph Storage Troubleshooting Guide* for more details.

5.5. REPLACING AN OSD DRIVE

Ceph is designed for fault tolerance, which means that it can operate in a **degraded** state without losing data. Consequently, Ceph can operate even if a data storage drive fails. In the context of a failed drive, the **degraded** state means that the extra copies of the data stored on other OSDs will backfill automatically to other OSDs in the cluster. However, if this occurs, replace the failed OSD drive and recreate the OSD manually.

When a drive fails, Ceph reports the OSD as **down**:

```
HEALTH_WARN 1/3 in osds are down
osd.0 is down since epoch 23, last address 192.168.106.220:6800/11080
```

NOTE

Ceph can mark an OSD as **down** also as a consequence of networking or permissions problems. See *Down OSDs* for details.

Modern servers typically deploy with hot-swappable drives so you can pull a failed drive and replace it with a new one without bringing down the node. The whole procedure includes these steps:

1. Remove the OSD from the Ceph cluster. For details, see the *Removing an OSD from the Ceph Cluster* procedure.
2. Replace the drive. For details, see *Replacing the physical drive*] section.
3. Add the OSD to the cluster. For details, see *Adding an OSD to the Ceph Cluster]* procedure.

Prerequisites

- Root-level access to the Ceph Monitor node.
- Determine which OSD is **down**:

```
[root@mon ~]# ceph osd tree | grep -i down
ID WEIGHT  TYPE NAME      UP/DOWN REWEIGHT PRIMARY-AFFINITY
 0 0.00999         osd.0     down  1.00000          1.00000
```

- Ensure that the OSD process is stopped. Use the following command from the OSD node:

```
[root@mon ~]# systemctl status ceph-osd@_OSD_NUMBER_
```

- Replace ***OSD_NUMBER*** with the ID of the OSD marked as **down**, for example:

```
[root@mon ~]# systemctl status ceph-osd@osd.0
...
   Active: inactive (dead)
```

If the **ceph-osd** daemon is running. See *Down OSDs* for more details about troubleshooting OSDs that are marked as **down** but their corresponding **ceph-osd** daemon is running.

Procedure: Removing an OSD from the Ceph Cluster

1. Mark the OSD as **out**:

```
[root@mon ~]# ceph osd out osd.OSD_NUMBER
```

Replace ***OSD_NUMBER*** with the ID of the OSD that is marked as **down**, for example:

```
[root@mon ~]# ceph osd out osd.0
marked out osd.0.
```

NOTE

If the OSD is **down**, Ceph marks it as **out** automatically after 600 seconds when it does not receive any heartbeat packet from the OSD. When this happens, other OSDs with copies of the failed OSD data begin backfilling to ensure that the required number of copies exists within the cluster. While the cluster is backfilling, the cluster will be in a **degraded** state.

2. Ensure that the failed OSD is backfilling. The output will include information similar to the following one:

```
[root@mon ~]# ceph -w | grep backfill
2017-06-02 04:48:03.403872 mon.0 [INF] pgmap v10293282: 431 pgs: 1
active+undersized+degraded+remapped+backfilling, 28 active+undersized+degraded, 49
active+undersized+degraded+remapped+wait_backfill, 59 stale+active+clean, 294
active+clean; 72347 MB data, 101302 MB used, 1624 GB / 1722 GB avail; 227 kB/s rd, 1358
B/s wr, 12 op/s; 10626/35917 objects degraded (29.585%); 6757/35917 objects misplaced
(18.813%); 63500 kB/s, 15 objects/s recovering
2017-06-02 04:48:04.414397 mon.0 [INF] pgmap v10293283: 431 pgs: 2
active+undersized+degraded+remapped+backfilling, 75
active+undersized+degraded+remapped+wait_backfill, 59 stale+active+clean, 295
active+clean; 72347 MB data, 101398 MB used, 1623 GB / 1722 GB avail; 969 kB/s rd, 6778
B/s wr, 32 op/s; 10626/35917 objects degraded (29.585%); 10580/35917 objects misplaced
(29.457%); 125 MB/s, 31 objects/s recovering
2017-06-02 04:48:00.380063 osd.1 [INF] 0.6f starting backfill to osd.0 from (0'0,0'0] MAX to
2521'166639
2017-06-02 04:48:00.380139 osd.1 [INF] 0.48 starting backfill to osd.0 from (0'0,0'0] MAX to
2513'43079
2017-06-02 04:48:00.380260 osd.1 [INF] 0.d starting backfill to osd.0 from (0'0,0'0] MAX to
2513'136847
2017-06-02 04:48:00.380849 osd.1 [INF] 0.71 starting backfill to osd.0 from (0'0,0'0] MAX to
2331'28496
2017-06-02 04:48:00.381027 osd.1 [INF] 0.51 starting backfill to osd.0 from (0'0,0'0] MAX to
2513'87544
```

3. Remove the OSD from the CRUSH map:

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

Replace ***OSD_NUMBER*** with the ID of the OSD that is marked as **down**, for example:

```
[root@mon ~]# ceph osd crush remove osd.0
removed item id 0 name 'osd.0' from crush map
```

4. Remove authentication keys related to the OSD:

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

Replace ***OSD_NUMBER*** with the ID of the OSD that is marked as **down**, for example:

```
[root@mon ~]# ceph auth del osd.0
updated
```

5. Remove the OSD from the Ceph Storage Cluster:

```
[root@mon ~]# ceph osd rm osd.OSD_NUMBER
```

Replace ***OSD_NUMBER*** with the ID of the OSD that is marked as **down**, for example:

```
[root@mon ~]# ceph osd rm osd.0
removed osd.0
```

If you have removed the OSD successfully, it is not present in the output of the following command:

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

6. For bare-metal deployments, unmount the failed drive:

```
[root@mon ~]# umount /var/lib/ceph/osd/CLUSTER_NAME-OSD_NUMBER
```

Specify the name of the cluster and the ID of the OSD, for example:

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

If you have unmounted the drive successfully, it is not present in the output of the following command:

```
[root@mon ~]# df -h
```

Procedure: Replacing the physical drive

See the documentation for the hardware node for details on replacing the physical drive.

a. If the drive is hot-swappable, replace the failed drive with a new one.

b. If the drive is not hot-swappable and the node contains multiple OSDs, you might have to shut down the whole node and replace the physical drive. Consider preventing the cluster from backfilling. See the *Stopping and Starting Rebalancing* chapter in the *Red Hat Ceph Storage Troubleshooting Guide* for details.

c. When the drive appears under the /**dev**/ directory, make a note of the drive path.

d. If you want to add the OSD manually, find the OSD drive and format the disk.

Procedure: Adding an OSD to the Ceph Cluster

1. Add the OSD again.

   a. If you used Ansible to deploy the cluster, run the **ceph-ansible** playbook again from the Ceph administration server:

   - **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
     ```

   b. If you added the OSD manually, see the *Adding a Ceph OSD with the Command-line Interface* section in the *Red Hat Ceph Storage 4 Operations Guide*.

2. Ensure that the CRUSH hierarchy is accurate:

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

3. If you are not satisfied with the location of the OSD in the CRUSH hierarchy, move the OSD to a desired location:

   ```
   [root@mon ~]# ceph osd crush move BUCKET_TO_MOVE
   BUCKET_TYPE=PARENT_BUCKET
   ```

   For example, to move the bucket located at **sdd:row1** to the root bucket:

   ```
   [root@mon ~]# ceph osd crush move ssd:row1 root=ssd:root
   ```

Additional Resources

- See the *Down OSDs* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Managing the storage cluster size* chapter in the *Red Hat Ceph Storage Operations Guide*.
- See the *Red Hat Ceph Storage Installation Guide*.

5.6. INCREASING THE PID COUNT

If you have a node containing more than 12 Ceph OSDs, the default maximum number of threads (PID count) can be insufficient, especially during recovery. As a consequence, some **ceph-osd** daemons can terminate and fail to start again. If this happens, increase the maximum possible number of threads allowed.

Procedure

To temporary increase the number:

```
[root@mon ~]# sysctl -w kernel.pid.max=4194303
```

To permanently increase the number, update the **/etc/sysctl.conf** file as follows:

```
kernel.pid.max = 4194303
```

5.7. DELETING DATA FROM A FULL STORAGE CLUSTER

Ceph automatically prevents any I/O operations on OSDs that reached the capacity specified by the **mon_osd_full_ratio** parameter and returns the **full osds** error message.

This procedure shows how to delete unnecessary data to fix this error.

NOTE

The **mon_osd_full_ratio** parameter sets the value of the **full_ratio** parameter when creating a cluster. You cannot change the value of **mon_osd_full_ratio** afterwards. To temporarily increase the **full_ratio** value, increase the **set-full-ratio** instead.

Prerequisites

- Root-level access to the Ceph Monitor node.

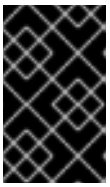
Procedure

1. Determine the current value of **full_ratio**, by default it is set to **0.95**:

   ```
   [root@mon ~]# ceph osd dump | grep -i full
   full_ratio 0.95
   ```

2. Temporarily increase the value of **set-full-ratio** to **0.97**:

   ```
   [root@mon ~]# ceph osd set-full-ratio 0.97
   ```

   IMPORTANT

   Red Hat strongly recommends to not set the **set-full-ratio** to a value higher than 0.97. Setting this parameter to a higher value makes the recovery process harder. As a consequence, you might not be able to recover full OSDs at all.

3. Verify that you successfully set the parameter to **0.97**:

```
[root@mon ~]# ceph osd dump | grep -i full
full_ratio 0.97
```

4. Monitor the cluster state:

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

As soon as the cluster changes its state from **full** to **nearfull**, delete any unnecessary data.

5. Set the value of **full_ratio** back to **0.95**:

```
[root@mon ~]# ceph osd set-full-ratio 0.95
```

6. Verify that you successfully set the parameter to **0.95**:

```
[root@mon ~]# ceph osd dump | grep -i full
full_ratio 0.95
```

Additional Resources

- *Full OSDs* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- *Nearfull OSDs* section in the *Red Hat Ceph Storage Troubleshooting Guide*.

5.8. REDEPLOYING OSDS AFTER UPGRADING THE STORAGE CLUSTER

This section describes how to redeploy OSDS after upgrading from Red Hat Ceph Storage 3 to Red Hat Ceph Storage 4 with non-collocated daemons for OSDs with **block.db** on dedicated devices, without upgrading the operating system.

This procedure applies to both bare-metal and container deployments, unless specified.

After the upgrade, the playbook for redeploying OSDs can fail with the an error message:

```
GPT headers found, they must be removed on: /dev/vdb
```

You can redeploy the OSDs by creating a partition in the **block.db** device and running the Ansible playbook.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ansible Administration node.
- Ansible user account created.

Procedure

1. Create the partition on the **block.db** device. This **sgdisk** command uses the next available partition number automatically:

Syntax

```
sgdisk --new=0:0:_JOURNAL_SIZE_ -- NEW_DEVICE_PATH
```

Example

```
[root@admin ~]# sgdisk --new=0:0:+2G -- /dev/vdb
```

2. Create the **host_vars** directory:

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

3. Navigate to the **host_vars** directory:

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

4. Create the hosts file on all the hosts of the storage cluster:

Syntax

```
touch NEW_OSD_HOST_NAME
```

Example

```
[root@admin host_vars]# touch osd5
```

5. In the hosts file, define the data device:

Syntax

```
lvm_volumes:
- data: DATA_DEVICE_PATH
  journal: NEW_DEVICE_PARTITION_PATH
- data: RUNNING_OSD_DATA_DEVICE_PATH
    journal: PARTITION_PATH
- data: RUNNING_OSD_DATA_DEVICE_PATH
    journal: PARTITION_PATH
```

Example

```
lvm_volumes:
- data: /dev/vdd
  journal: /dev/vdb2
- data: /dev/sdb
  journal: /dev/sdc1
- data: /dev/sdb
  journal: /dev/sdc2
```

6. Switch to the Ansible user and verify that Ansible can reach all the Ceph nodes:

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

7. Change directory to the Ansible configuration directory:

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

8. Run the following Ansible playbook with **--limit** option:

   - **Bare-metal** deployments:

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

   - **Container** deployments:

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

Additional Resources

- See the *Handling a disk failure* section in the *Red Hat Ceph Storage Operations Guide* for more details on deploying OSDs.

CHAPTER 6. TROUBLESHOOTING CEPH MDSS

As a storage administrator, you can troubleshoot the most common issues that can occur when using the Ceph Metadata Server (MDS). Some of the common errors that you might encounter:

- An MDS node failure requiring a new MDS deployment.
- An MDS node issue requiring redeployment of an MDS node.

6.1. REDEPLOYING A CEPH MDS

Ceph Metadata Server (MDS) daemons are necessary for deploying a Ceph File System. If an MDS node in your cluster fails, you can redeploy a Ceph Metadata Server by removing an MDS server and adding a new or existing server. You can use the command-line interface or Ansible playbook to add or remove an MDS server.

6.1.1. Prerequisites

- A running Red Hat Ceph Storage cluster.

6.1.2. 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.

6.1.3. 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.

6.1.4. 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.

6.1.5. 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.

CHAPTER 7. TROUBLESHOOTING A MULTISITE CEPH OBJECT GATEWAY

This chapter contains information on how to fix the most common errors related to multisite Ceph Object Gateways configuration and operational conditions.

7.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.
- A running Ceph Object Gateway.

7.2. ERROR CODE DEFINITIONS FOR THE CEPH OBJECT GATEWAY

The Ceph Object Gateway logs contain error and warning messages to assist in troubleshooting conditions in your environment. Some common ones are listed below with suggested resolutions.

Common error messages

data_sync: ERROR: a sync operation returned error

This is the high-level data sync process complaining that a lower-level bucket sync process returned an error. This message is redundant; the bucket sync error appears above it in the log.

data sync: ERROR: failed to sync object: *BUCKET_NAME*:_OBJECT_NAME_

Either the process failed to fetch the required object over HTTP from a remote gateway or the process failed to write that object to RADOS and it will be tried again.

data sync: ERROR: failure in sync, backing out (sync_status=2)

A low level message reflecting one of the above conditions, specifically that the data was deleted before it could sync and thus showing a **-2 ENOENT** status.

data sync: ERROR: failure in sync, backing out (sync_status=-5)

A low level message reflecting one of the above conditions, specifically that we failed to write that object to RADOS and thus showing a **-5 EIO**.

ERROR: failed to fetch remote data log info: ret=11

This is the **EAGAIN** generic error code from **libcurl** reflecting an error condition from another gateway. It will try again by default.

meta sync: ERROR: failed to read mdlog info with (2) No such file or directory

The shard of the mdlog was never created so there is nothing to sync.

Syncing error messages

failed to sync object

Either the process failed to fetch this object over HTTP from a remote gateway or it failed to write that object to RADOS and it will be tried again.

failed to sync bucket instance: (11) Resource temporarily unavailable

A connection issue between primary and secondary zones.

failed to sync bucket instance: (125) Operation canceled

A racing condition exists between writes to the same RADOS object.

Additional Resources

- Contact Red Hat Support for any additional assistance.

7.3. SYNCING A MULTISITE CEPH OBJECT GATEWAY

A multisite sync reads the change log from other zones. To get a high-level view of the sync progress from the metadata and the data loags, you can use the following command:

```
radosgw-admin sync status
```

This command lists which log shards, if any, which are behind their source zone.

NOTE

Sometimes you might observe recovering shards when running the **radosgw-admin sync status** command. For data sync, there are 128 shards of replication logs that are each processed independently. If any of the actions triggered by these replication log events result in any error from the network, storage, or elsewhere, those errors get tracked so the operation can retry again later. While a given shard has errors that need a retry, **radosgw-admin sync status** command reports that shard as **recovering**. This recovery happens automatically, so the operator does not need to intervene to resolve them.

If the results of the sync status you have run above reports log shards are behind, run the following command substituting the shard-id for *X*.

Syntax

```
radosgw-admin data sync status --shard-id=X --source-zone=ZONE_NAME
```

Example

```
[root@rgw ~]# radosgw-admin data sync status --shard-id=27 --source-zone=_us-eest
{
  "shard_id": 27,
  "marker": {
        "status": "incremental-sync",
        "marker": "1_1534494893.816775_131867195.1",
        "next_step_marker": "",
        "total_entries": 1,
        "pos": 0,
        "timestamp": "0.000000"
   },
   "pending_buckets": [],
   "recovering_buckets": [
        "pro-registry:4ed07bb2-a80b-4c69-aa15-fdc17ae6f5f2.314303.1:26"
   ]
}
```

The output lists which buckets are next to sync and which buckets, if any, are going to be retried due to previous errors.

Inspect the status of individual buckets with the following command, substituting the bucket id for *X*.

```
radosgw-admin bucket sync status --bucket=X.
```

Replace...

X with the ID number of the bucket.

The result shows which bucket index log shards are behind their source zone.

A common error in sync is **EBUSY**, which means the sync is already in progress, often on another gateway. Read errors written to the sync error log, which can be read with the following command:

```
radosgw-admin sync error list
```

The syncing process will try again until it is successful. Errors can still occur that can require intervention.

7.3.1. Performance counters for multi-site Ceph Object Gateway data sync

The following performance counters are available for multi-site configurations of the Ceph Object Gateway to measure data sync:

- **poll_latency** measures the latency of requests for remote replication logs.
- **fetch_bytes** measures the number of objects and bytes fetched by data sync.

Use the **ceph daemon** command to view the current metric data for the performance counters:

Syntax

```
ceph daemon /var/run/ceph/ceph-client.rgw.RGW_ID.asok perf dump data-sync-from-ZONE_NAME
```

Example

```
[root@rgw ~]#  ceph daemon /var/run/ceph/ceph-client.rgw.host02-rgw0.103.94309060818504.asok
perf dump data-sync-from-us-west

{
    "data-sync-from-us-west": {
        "fetch bytes": {
            "avgcount": 54,
            "sum": 54526039885
        },
        "fetch not modified": 7,
        "fetch errors": 0,
        "poll latency": {
            "avgcount": 41,
            "sum": 2.533653367,
            "avgtime": 0.061796423
        },
        "poll errors": 0
    }
}
```

NOTE

You must run the **ceph daemon** command from the node running the daemon.

Additional Resources

- See the *Ceph performance counters* chapter in the *Red Hat Ceph Storage Administration Guide* for more information about performance counters.

CHAPTER 8. TROUBLESHOOTING THE CEPH ISCSI GATEWAY (LIMITED AVAILABILITY)

As a storage administrator, you can troubleshoot most common errors that can occur when using the Ceph iSCSI gateway. These are some of the common errors that you might encounter:

- iSCSI login issues.
- VMware ESXi reporting various connection failures.
- Timeout errors.

NOTE

This technology is Limited Availability. See the *Deprecated functionality* chapter for additional information.

8.1. PREREQUISITES

- A running Red Hat Ceph Storage cluster.
- A running Ceph iSCSI gateway.
- Verify the network connections.

8.2. GATHERING INFORMATION FOR LOST CONNECTIONS CAUSING STORAGE FAILURES ON VMWARE ESXI

Collecting system and disk information helps determine which iSCSI target has lost a connection and is possibly causing storage failures. If needed, gathering this information can also be provided to Red Hat's Global Support Service to aid you in troubleshooting any Ceph iSCSI gateway issues.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A running Ceph iSCSI gateway, the iSCSI target.
- A running VMware ESXi environment, the iSCSI initiator.
- Root-level access to the VMware ESXi node.

Procedure

1. On the VWware ESXi node, open the kernel log:

```
[root@esx:~]# more /var/log/vmkernel.log
```

2. Gather information from the following error messages in the VMware ESXi kernel log:

   Example

```
2020-03-30T11:07:07.570Z cpu32:66506)iscsi_vmk:
iscsivmk_ConnRxNotifyFailure: Sess [ISID: 00023d000005 TARGET:
iqn.2017-12.com.redhat.iscsi-gw:ceph-igw TPGT: 3 TSIH: 0]
```

From this message, make a note of the **ISID** number, the **TARGET** name, and the Target Portal Group Tag (**TPGT**) number. For this example, we have the following:

```
ISID: 00023d000005
TARGET: iqn.2017-12.com.redhat.iscsi-gw:ceph-igw
TPGT: 3
```

Example

```
2020-03-30T11:07:07.570Z cpu32:66506)iscsi_vmk:
iscsivmk_ConnRxNotifyFailure: vmhba64:CH:4 T:0 CN:0: Connection rx
notifying failure: Failed to Receive. State=Bound
```

From this message, make a note of the adapter channel (**CH**) number. For this example, we have the following:

```
vmhba64:CH:4 T:0
```

3. To find the remote address of the Ceph iSCSI gateway node:

```
[root@esx:~]# esxcli iscsi session connection list
```

Example

```
...
vmhba64,iqn.2017-12.com.redhat.iscsi-gw:ceph-igw,00023d000003,0
   Adapter: vmhba64
   Target: iqn.2017-12.com.redhat.iscsi-gw:ceph-igw (1)
   ISID: 00023d000003 (2)
   CID: 0
   DataDigest: NONE
   HeaderDigest: NONE
   IFMarker: false
   IFMarkerInterval: 0
   MaxRecvDataSegmentLength: 131072
   MaxTransmitDataSegmentLength: 262144
   OFMarker: false
   OFMarkerInterval: 0
   ConnectionAddress: 10.2.132.2
   RemoteAddress: 10.2.132.2 (3)
   LocalAddress: 10.2.128.77
   SessionCreateTime: 03/28/18 21:45:19
   ConnectionCreateTime: 03/28/18 21:45:19
   ConnectionStartTime: 03/28/18 21:45:19
   State: xpt_wait
...
```

From the command output, match the **ISID** value, and the **TARGET** name value gathered previously, then make a note of the **RemoteAddress** value. From this example, we have the following:

```
Target: iqn.2017-12.com.redhat.iscsi-gw:ceph-igw
ISID: 00023d000003
RemoteAddress: 10.2.132.2
```

Now, you can collect more information from the Ceph iSCSI gateway node to further troubleshoot the issue.

a. On the Ceph iSCSI gateway node mentioned by the **RemoteAddress** value, run an **sosreport** to gather system information:

```
[root@igw ~]# sosreport
```

4. To find a disk that went into a dead state:

```
[root@esx:~]# esxcli storage nmp device list
```

Example

```
...
iqn.1998-01.com.vmware:d04-nmgjd-pa-zyc-sv039-rh2288h-xnh-732d78fd-
00023d000004,iqn.2017-12.com.redhat.iscsi-gw:ceph-igw,t,3-
naa.60014054a5d46697f85498e9a257567c
   Runtime Name: vmhba64:C4:T0:L4 (1)
   Device: naa.60014054a5d46697f85498e9a257567c (2)
   Device Display Name: LIO-ORG iSCSI Disk
(naa.60014054a5d46697f85498e9a257567c)
   Group State: dead (3)
   Array Priority: 0
   Storage Array Type Path Config:
{TPG_id=3,TPG_state=ANO,RTP_id=3,RTP_health=DOWN} (4)
   Path Selection Policy Path Config: {non-current path; rank: 0}
...
```

From the command output, match the **CH** number, and the **TPGT** number gathered previously, then make a note of the **Device** value. For this example, we have the following:

```
vmhba64:C4:T0
Device: naa.60014054a5d46697f85498e9a257567c
TPG_id=3
```

With the device name, you can gather some additional information on each iSCSI disk in a **dead** state.

a. Gather more information on the iSCSI disk:

Syntax

```
esxcli storage nmp path list -d ISCSI_DISK_DEVICE >
/tmp/esxcli_storage_nmp_path_list.txt
```

```
esxcli storage core device list -d ISCSI_DISK_DEVICE >
/tmp/esxcli_storage_core_device_list.txt
```

Example

```
[root@esx:~]# esxcli storage nmp path list -d naa.60014054a5d46697f85498e9a257567c
> /tmp/esxcli_storage_nmp_path_list.txt
[root@esx:~]# esxcli storage core device list -d
naa.60014054a5d46697f85498e9a257567c > /tmp/esxcli_storage_core_device_list.txt
```

5. Gather additional information on the VMware ESXi environment:

```
[root@esx:~]# esxcli storage vmfs extent list > /tmp/esxcli_storage_vmfs_extent_list.txt
[root@esx:~]# esxcli storage filesystem list > /tmp/esxcli_storage_filesystem_list.txt
[root@esx:~]# esxcli iscsi session list > /tmp/esxcli_iscsi_session_list.txt
[root@esx:~]# esxcli iscsi session connection list >
/tmp/esxcli_iscsi_session_connection_list.txt
```

6. Check for potential iSCSI login issues:
   - Was the iSCSI login data not sent?
   - Did the iSCSI login timeout or fail to find a portal group?

Additional Resources

- See Red Hat's Knowledgebase solution on creating an **sosreport** for Red Hat Global Support Services.
- See Red Hat's Knowledgebase solution on uploading files for Red Hat Global Support Services.
- How to open a Red Hat support case on the Customer Portal?

8.3. CHECKING ISCSI LOGIN FAILURES BECAUSE DATA WAS NOT SENT

On the iSCSI gateway node, you might see generic login negotiation failure messages in the system log, by default **/var/log/messages**.

Example

```
Apr  2 23:17:05 osd1 kernel: rx_data returned 0, expecting 48.
Apr  2 23:17:05 osd1 kernel: iSCSI Login negotiation failed.
```

While the system is in this state, start collecting system information as suggested in this procedure.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- A running Ceph iSCSI gateway, the iSCSI target.
- A running VMware ESXi environment, the iSCSI initiator.

- Root-level access to the Ceph iSCSI gateway node.
- Root-level access to the VMware ESXi node.

Procedure

1. Enable additional logging:

```
[root@igw ~]# echo "module iscsi_target_mod +p" >
/sys/kernel/debug/dynamic_debug/control
[root@igw ~]# echo "module target_core_mod +p" >
/sys/kernel/debug/dynamic_debug/control
```

2. Wait a couple of minutes for the extra debugging information to populate the system log.
3. Disable the additional logging:

```
[root@igw ~]# echo "module iscsi_target_mod -p" >
/sys/kernel/debug/dynamic_debug/control
[root@igw ~]# echo "module target_core_mod -p" >
/sys/kernel/debug/dynamic_debug/control
```

4. Run an **sosreport** to gather system information:

```
[root@igw ~]# sosreport
```

5. Capture network traffic for the Ceph iSCSI gateway and the VMware ESXi nodes simultaneously:

   Syntax

```
tcpdump -s0 -i NETWORK_INTERFACE -w OUTPUT_FILE_PATH
```

   Example

```
[root@igw ~]# tcpdump -s 0 -i eth0 -w /tmp/igw-eth0-tcpdump.pcap
```

NOTE

Look for traffic on port 3260.

   a. Network packet capture files can be large, so compress the **tcpdump** output from the iSCSI target and initiators before uploading any files to Red Hat Global Support Services:

      Syntax

```
gzip OUTPUT_FILE_PATH
```

      Example

```
[root@igw ~]# gzip /tmp/igw-eth0-tcpdump.pcap
```

6. Gather additional information on the VMware ESXi environment:

```
[root@esx:~]# esxcli iscsi session list > /tmp/esxcli_iscsi_session_list.txt
[root@esx:~]# esxcli iscsi session connection list >
/tmp/esxcli_iscsi_session_connection_list.txt
```

   a. List and collect more information on each iSCSI disk:

Syntax

```
esxcli storage nmp path list -d ISCSI_DISK_DEVICE >
/tmp/esxcli_storage_nmp_path_list.txt
```

Example

```
[root@esx:~]# esxcli storage nmp device list
[root@esx:~]# esxcli storage nmp path list -d naa.60014054a5d46697f85498e9a257567c
> /tmp/esxcli_storage_nmp_path_list.txt
[root@esx:~]# esxcli storage core device list -d
naa.60014054a5d46697f85498e9a257567c > /tmp/esxcli_storage_core_device_list.txt
```

Additional Resources

- See Red Hat's Knowledgebase solution on creating an **sosreport** for Red Hat Global Support Services.
- See Red Hat's Knowledgebase solution on uploading files for Red Hat Global Support Services.
- See Red Hat's Knowledgebase solution on How to capture network packets with tcpdump? for more information.
- How to open a Red Hat support case on the Customer Portal?

8.4. CHECKING ISCSI LOGIN FAILURES BECAUSE OF A TIMEOUT OR NOT ABLE TO FIND A PORTAL GROUP

On the iSCSI gateway node, you might see timeout or unable to locate a target portal group messages in the system log, by default **/var/log/messages**.

Example

```
Mar 28 00:29:01 osd2 kernel: iSCSI Login timeout on Network Portal 10.2.132.2:3260
```

or

Example

```
Mar 23 20:25:39 osd1 kernel: Unable to locate Target Portal Group on iqn.2017-12.com.redhat.iscsi-gw:ceph-igw
```

While the system is in this state, start collecting system information as suggested in this procedure.

Prerequisites

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

Procedure

1. Enable the dumping of waiting tasks and write them to a file:

```
[root@igw ~]# dmesg -c ; echo w > /proc/sysrq-trigger ; dmesg -c > /tmp/waiting-tasks.txt
```

2. Review the list of waiting tasks for the following messages:

   - **iscsit_tpg_disable_portal_group**
   - **core_tmr_abort_task**
   - **transport_generic_free_cmd**

   If any of these messages appear in the waiting task list, then this is an indication that something went wrong with the **tcmu-runner** service. Maybe the **tcmu-runner** service was not restarted properly, or maybe the **tcmu-runner** service has crashed.

3. Verify if the **tcmu-runner** service is running:

```
[root@igw ~]# systemctl status tcmu-runner
```

   a. If the **tcmu-runner** service is not running, then stop the **rbd-target-gw** service before restarting the **tcmu-runner** service:

```
[root@igw ~]# systemctl stop rbd-target-api
[root@igw ~]# systemctl stop rbd-target-gw
[root@igw ~]# systemctl stop tcmu-runner
[root@igw ~]# systemctl start tcmu-runner
[root@igw ~]# systemctl start rbd-target-gw
[root@igw ~]# systemctl start rbd-target-api
```

IMPORTANT

Stopping the Ceph iSCSI gateway first prevents IOs from getting stuck while the **tcmu-runner** service is down.

   b. If the **tcmu-runner** service is running, the this might be a new bug. Open a new Red Hat support case.

Additional Resources

- See Red Hat's Knowledgebase solution on creating an **sosreport** for Red Hat Global Support Services.
- See Red Hat's Knowledgebase solution on uploading files for Red Hat Global Support Services.

- How to open a Red Hat support case on the Customer Portal?

8.5. TIMEOUT COMMAND ERRORS

The Ceph iSCSI gateway might report command timeout errors when a SCSI command has failed in the system log.

Example

```
Mar 23 20:03:14 igw tcmu-runner: 2018-03-23 20:03:14.052 2513 [ERROR]
tcmu_rbd_handle_timedout_cmd:669 rbd/rbd.gw1lun011: Timing out cmd.
```

or

Example

```
Mar 23 20:03:14 igw tcmu-runner: tcmu_notify_conn_lost:176 rbd/rbd.gw1lun011: Handler
connection lost (lock state 1)
```

What This Means

It is possible there are other stuck tasks waiting to be processed, causing the SCSI command to timeout because a response was not received in a timely manner. Another reason for these error messages, might be related to an unhealthy Red Hat Ceph Storage cluster.

To Troubleshoot This Problem

1. Check to see if there are waiting tasks that might be holding things up.
2. Check the health of the Red Hat Ceph Storage cluster.
3. Collect system information from each device in the path from the Ceph iSCSI gateway node to the iSCSI initiator node.

Additional Resources

- See the *Checking iSCSI login failures because of a timeout or not able to find a portal group* section of the *Red Hat Ceph Storage Troubleshooting Guide* for more details on how to view waiting tasks.
- See the *Diagnosing the health of a storage cluster* section of the *Red Hat Ceph Storage Troubleshooting Guide* for more details on checking the storage cluster health.
- See the *Gathering information for lost connections causing storage failures on VMware ESXi* section of the *Red Hat Ceph Storage Troubleshooting Guide* for more details on collecting the necessary information.

8.6. ABORT TASK ERRORS

The Ceph iSCSI gateway might report abort task errors in the system log.

Example

```
Apr  1 14:23:58 igw kernel: ABORT_TASK: Found referenced iSCSI task_tag: 1085531
```

-

What This Means

It is possible that some other network disruptions, such as a failed switch or bad port, is causing this type of error message. Another possibility, is an unhealthy Red Hat Ceph Storage cluster.

To Troubleshoot This Problem

1. Check for any network disruptions in the environment.
2. Check the health of the Red Hat Ceph Storage cluster.
3. Collect system information from each device in the path from the Ceph iSCSI gateway node to the iSCSI initiator node.

Additional Resources

- See the *Diagnosing the health of a storage cluster* section of the *Red Hat Ceph Storage Troubleshooting Guide* for more details on checking the storage cluster health.
- See the *Gathering information for lost connections causing storage failures on VMware ESXi* section of the *Red Hat Ceph Storage Troubleshooting Guide* for more details on collecting the necessary information.

8.7. ADDITIONAL RESOURCES

- See the *Red Hat Ceph Storage Block Device Guide* for more details on the Ceph iSCSI gateway.
- See Chapter 3, *Troubleshooting networking issues* for details.

CHAPTER 9. TROUBLESHOOTING CEPH PLACEMENT GROUPS

This section contains information about fixing the most common errors related to the Ceph Placement Groups (PGs).

9.1. PREREQUISITES

- Verify your network connection.
- Ensure that Monitors are able to form a quorum.
- Ensure that all healthy OSDs are **up** and **in**, and the backfilling and recovery processes are finished.

9.2. MOST COMMON CEPH PLACEMENT GROUPS ERRORS

The following table lists the most common errors messages that are returned by the **ceph health detail** command. The table provides links to corresponding sections that explain the errors and point to specific procedures to fix the problems.

In addition, you can list placement groups that are stuck in a state that is not optimal. See Section 9.3, "Listing placement groups stuck in **stale**, **inactive**, or **unclean** state" for details.

9.2.1. Prerequisites

- A running Red Hat Ceph Storage cluster.
- A running Ceph Object Gateway.

9.2.2. Placement group error messages

A table of common placement group error messages, and a potential fix.

Error message	See
HEALTH_ERR	
pgs down	Placement groups are **down**
pgs inconsistent	Inconsistent placement groups
scrub errors	Inconsistent placement groups
HEALTH_WARN	
pgs stale	Stale placement groups
unfound	Unfound objects

9.2.3. Stale placement groups

The **ceph health** command lists some Placement Groups (PGs) as **stale**:

```
HEALTH_WARN 24 pgs stale; 3/300 in osds are down
```

What This Means

The Monitor marks a placement group as **stale** when it does not receive any status update from the primary OSD of the placement group's acting set or when other OSDs reported that the primary OSD is **down**.

Usually, PGs enter the **stale** state after you start the storage cluster and until the peering process completes. However, when the PGs remain **stale** for longer than expected, it might indicate that the primary OSD for those PGs is **down** or not reporting PG statistics to the Monitor. When the primary OSD storing **stale** PGs is back **up**, Ceph starts to recover the PGs.

The **mon_osd_report_timeout** setting determines how often OSDs report PGs statistics to Monitors. Be default, this parameter is set to **0.5**, which means that OSDs report the statistics every half a second.

To Troubleshoot This Problem

1. Identify which PGs are **stale** and on what OSDs they are stored. The error message will include information similar to the following example:

   Example

   ```
   # ceph health detail
   HEALTH_WARN 24 pgs stale; 3/300 in osds are down
   ...
   pg 2.5 is stuck stale+active+remapped, last acting [2,0]
   ...
   osd.10 is down since epoch 23, last address 192.168.106.220:6800/11080
   osd.11 is down since epoch 13, last address 192.168.106.220:6803/11539
   osd.12 is down since epoch 24, last address 192.168.106.220:6806/11861
   ```

2. Troubleshoot any problems with the OSDs that are marked as **down**. For details, see *Down OSDs*.

Additional Resources

- The *Monitoring Placement Group sets* section in the *Administration Guide* for Red Hat Ceph Storage 4

9.2.4. Inconsistent placement groups

Some placement groups are marked as **active + clean + inconsistent** and the **ceph health detail** returns an error messages similar to the following one:

```
HEALTH_ERR 1 pgs inconsistent; 2 scrub errors
pg 0.6 is active+clean+inconsistent, acting [0,1,2]
2 scrub errors
```

What This Means

When Ceph detects inconsistencies in one or more replicas of an object in a placement group, it marks the placement group as **inconsistent**. The most common inconsistencies are:

- Objects have an incorrect size.
- Objects are missing from one replica after a recovery finished.

In most cases, errors during scrubbing cause inconsistency within placement groups.

To Troubleshoot This Problem

1. Determine which placement group is in the **inconsistent** state:

```
# ceph health detail
HEALTH_ERR 1 pgs inconsistent; 2 scrub errors
pg 0.6 is active+clean+inconsistent, acting [0,1,2]
2 scrub errors
```

2. Determine why the placement group is **inconsistent**.

   a. Start the deep scrubbing process on the placement group:

```
[root@mon ~]# ceph pg deep-scrub ID
```

   Replace ***ID*** with the ID of the **inconsistent** placement group, for example:

```
[root@mon ~]# ceph pg deep-scrub 0.6
instructing pg 0.6 on osd.0 to deep-scrub
```

   b. Search the output of the **ceph -w** for any messages related to that placement group:

```
ceph -w | grep ID
```

   Replace ***ID*** with the ID of the **inconsistent** placement group, for example:

```
[root@mon ~]# ceph -w | grep 0.6
2015-02-26 01:35:36.778215 osd.106 [ERR] 0.6 deep-scrub stat mismatch, got 636/635
objects, 0/0 clones, 0/0 dirty, 0/0 omap, 0/0 hit_set_archive, 0/0 whiteouts,
1855455/1854371 bytes.
2015-02-26 01:35:36.788334 osd.106 [ERR] 0.6 deep-scrub 1 errors
```

3. If the output includes any error messages similar to the following ones, you can repair the **inconsistent** placement group. See *Repairing inconsistent placement groups* for details.

```
PG.ID shard OSD: soid OBJECT missing attr , missing attr _ATTRIBUTE_TYPE
PG.ID shard OSD: soid OBJECT digest 0 != known digest DIGEST, size 0 != known size
SIZE
PG.ID shard OSD: soid OBJECT size 0 != known size SIZE
PG.ID deep-scrub stat mismatch, got MISMATCH
PG.ID shard OSD: soid OBJECT candidate had a read error, digest 0 != known digest
DIGEST
```

4. If the output includes any error messages similar to the following ones, it is not safe to repair the **inconsistent** placement group because you can lose data. Open a support ticket in this situation. See *Contacting Red Hat support* for details.

```
PG.ID shard OSD: soid OBJECT digest DIGEST != known digest DIGEST
PG.ID shard OSD: soid OBJECT omap_digest DIGEST != known omap_digest DIGEST
```

Additional Resources

- *Listing placement group inconsistencies* in the *Red Hat Ceph Storage Troubleshooting Guide*.
- The *Ceph Data integrity* section in the *Red Hat Ceph Storage Architecture Guide*.
- The *Scrubbing the OSD* section in the *Red Hat Ceph Storage Configuration Guide*.

9.2.5. Unclean placement groups

The **ceph health** command returns an error message similar to the following one:

```
HEALTH_WARN 197 pgs stuck unclean
```

What This Means

Ceph marks a placement group as **unclean** if it has not achieved the **active+clean** state for the number of seconds specified in the **mon_pg_stuck_threshold** parameter in the Ceph configuration file. The default value of **mon_pg_stuck_threshold** is **300** seconds.

If a placement group is **unclean**, it contains objects that are not replicated the number of times specified in the **osd_pool_default_size** parameter. The default value of **osd_pool_default_size** is **3**, which means that Ceph creates three replicas.

Usually, **unclean** placement groups indicate that some OSDs might be **down**.

To Troubleshoot This Problem

1. Determine which OSDs are **down**:

```
# ceph osd tree
```

2. Troubleshoot and fix any problems with the OSDs. See *Down OSDs* for details.

Additional Resources

- *Listing placement groups stuck in stale inactive or unclean state* .

9.2.6. Inactive placement groups

The **ceph health** command returns a error message similar to the following one:

```
HEALTH_WARN 197 pgs stuck inactive
```

What This Means

Ceph marks a placement group as **inactive** if it has not be active for the number of seconds specified in the **mon_pg_stuck_threshold** parameter in the Ceph configuration file. The default value of **mon_pg_stuck_threshold** is **300** seconds.

Usually, **inactive** placement groups indicate that some OSDs might be **down**.

To Troubleshoot This Problem

1. Determine which OSDs are **down**:

```
# ceph osd tree
```

2. Troubleshoot and fix any problems with the OSDs.

Additional Resources

- *Listing placement groups stuck in stale inactive or unclean state*
- See *Down OSDs* for details.

9.2.7. Placement groups are down

The **ceph health detail** command reports that some placement groups are **down**:

```
HEALTH_ERR 7 pgs degraded; 12 pgs down; 12 pgs peering; 1 pgs recovering; 6 pgs stuck
unclean; 114/3300 degraded (3.455%); 1/3 in osds are down
...
pg 0.5 is down+peering
pg 1.4 is down+peering
...
osd.1 is down since epoch 69, last address 192.168.106.220:6801/8651
```

What This Means

In certain cases, the peering process can be blocked, which prevents a placement group from becoming active and usable. Usually, a failure of an OSD causes the peering failures.

To Troubleshoot This Problem

Determine what blocks the peering process:

```
[root@mon ~]# ceph pg ID query
```

Replace ***ID*** with the ID of the placement group that is **down**, for example:

```
[root@mon ~]# ceph pg 0.5 query

{ "state": "down+peering",
  ...
  "recovery_state": [
      { "name": "Started\/Primary\/Peering\/GetInfo",
        "enter_time": "2012-03-06 14:40:16.169679",
        "requested_info_from": []},
      { "name": "Started\/Primary\/Peering",
        "enter_time": "2012-03-06 14:40:16.169659",
```

```
        "probing_osds": [
              0,
              1],
        "blocked": "peering is blocked due to down osds",
        "down_osds_we_would_probe": [
              1],
        "peering_blocked_by": [
              { "osd": 1,
                "current_lost_at": 0,
                "comment": "starting or marking this osd lost may let us proceed"}]},
      { "name": "Started",
        "enter_time": "2012-03-06 14:40:16.169513"}
  ]
}
```

The **recovery_state** section includes information why the peering process is blocked.

- If the output includes the **peering is blocked due to down osds** error message, see *Down OSDs*.
- If you see any other error message, open a support ticket. See *Contacting Red Hat Support service* for details.

Additional Resources

- The *Ceph OSD peering* section in the *Red Hat Ceph Storage Administration Guide*.

9.2.8. Unfound objects

The **ceph health** command returns an error message similar to the following one, containing the **unfound** keyword:

```
HEALTH_WARN 1 pgs degraded; 78/3778 unfound (2.065%)
```

What This Means

Ceph marks objects as **unfound** when it knows these objects or their newer copies exist but it is unable to find them. As a consequence, Ceph cannot recover such objects and proceed with the recovery process.

An Example Situation

A placement group stores data on **osd.1** and **osd.2**.

1. **osd.1** goes **down**.
2. **osd.2** handles some write operations.
3. **osd.1** comes **up**.
4. A peering process between **osd.1** and **osd.2** starts, and the objects missing on **osd.1** are queued for recovery.
5. Before Ceph copies new objects, **osd.2** goes **down**.

As a result, **osd.1** knows that these objects exist, but there is no OSD that has a copy of the objects.

In this scenario, Ceph is waiting for the failed node to be accessible again, and the **unfound** objects blocks the recovery process.

To Troubleshoot This Problem

1. Determine which placement group contain **unfound** objects:

```
[root@mon ~]# ceph health detail
HEALTH_WARN 1 pgs recovering; 1 pgs stuck unclean; recovery 5/937611 objects
degraded (0.001%); 1/312537 unfound (0.000%)
pg 3.8a5 is stuck unclean for 803946.712780, current state active+recovering, last acting
[320,248,0]
pg 3.8a5 is active+recovering, acting [320,248,0], 1 unfound
recovery 5/937611 objects degraded (0.001%); **1/312537 unfound (0.000%)**
```

2. List more information about the placement group:

```
[root@mon ~]# ceph pg ID query
```

Replace ***ID*** with the ID of the placement group containing the **unfound** objects, for example:

```
[root@mon ~]# ceph pg 3.8a5 query
{ "state": "active+recovering",
  "epoch": 10741,
  "up": [
        320,
        248,
        0],
  "acting": [
        320,
        248,
        0],
<snip>
  "recovery_state": [
        { "name": "Started\/Primary\/Active",
          "enter_time": "2015-01-28 19:30:12.058136",
          "might_have_unfound": [
                { "osd": "0",
                  "status": "already probed"},
                { "osd": "248",
                  "status": "already probed"},
                { "osd": "301",
                  "status": "already probed"},
                { "osd": "362",
                  "status": "already probed"},
                { "osd": "395",
                  "status": "already probed"},
                { "osd": "429",
                  "status": "osd is down"}],
          "recovery_progress": { "backfill_targets": [],
              "waiting_on_backfill": [],
              "last_backfill_started": "0\/\/0\/\/-1",
              "backfill_info": { "begin": "0\/\/0\/\/-1",
                  "end": "0\/\/0\/\/-1",
                  "objects": []},
```

```
            "peer_backfill_info": [],
            "backfills_in_flight": [],
            "recovering": [],
            "pg_backend": { "pull_from_peer": [],
                "pushing": []}},
          "scrub": { "scrubber.epoch_start": "0",
            "scrubber.active": 0,
            "scrubber.block_writes": 0,
            "scrubber.finalizing": 0,
            "scrubber.waiting_on": 0,
            "scrubber.waiting_on_whom": []}},
        { "name": "Started",
          "enter_time": "2015-01-28 19:30:11.044020"}],
```

The **might_have_unfound** section includes OSDs where Ceph tried to locate the **unfound** objects:

- The **already probed** status indicates that Ceph cannot locate the **unfound** objects in that OSD.
- The **osd is down** status indicates that Ceph cannot contact that OSD.

3. Troubleshoot the OSDs that are marked as **down**. See *Down OSDs* for details.
4. If you are unable to fix the problem that causes the OSD to be **down**, open a support ticket. See *Contacting Red Hat Support for service* for details.

9.3. LISTING PLACEMENT GROUPS STUCK IN STALE, INACTIVE, OR UNCLEAN STATE

After a failure, placement groups enter states like **degraded** or **peering**. This states indicate normal progression through the failure recovery process.

However, if a placement group stays in one of these states for a longer time than expected, it can be an indication of a larger problem. The Monitors reports when placement groups get stuck in a state that is not optimal.

The **mon_pg_stuck_threshold** option in the Ceph configuration file determines the number of seconds after which placement groups are considered **inactive**, **unclean**, or **stale**.

The following table lists these states together with a short explanation.

State	What it means	Most common causes	See
inactive	The PG has not been able to service read/write requests.	• Peering problems	Inactive placement groups

State	What it means	Most common causes	See
unclean	The PG contains objects that are not replicated the desired number of times. Something is preventing the PG from recovering.	• **unfound** objects • OSDs are **down** • Incorrect configuration	Unclean placement groups
stale	The status of the PG has not been updated by a **ceph-osd** daemon.	• OSDs are **down**	Stale placement groups

Prerequisites

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

Procedure

1. List the stuck PGs:

```
[root@mon ~]# ceph pg dump_stuck inactive
[root@mon ~]# ceph pg dump_stuck unclean
[root@mon ~]# ceph pg dump_stuck stale
```

Additional Resources

- See the *Placement group states* section in the *Red Hat Ceph Storage Administration Guide*.

9.4. LISTING PLACEMENT GROUP INCONSISTENCIES

Use the **rados** utility to list inconsistencies in various replicas of an objects. Use the **--format=json-pretty** option to list a more detailed output.

This section covers the listing of:

- *Inconsistent placement group in a pool*
- *Inconsistent objects in a placement group*
- *Inconsistent snapshot sets in a placement group*

Prerequisites

- A running Red Hat Ceph Storage cluster in a healthy state.
- Root-level access to the node.

Procedure

```
rados list-inconsistent-pg POOL --format=json-pretty
```

For example, list all inconsistent placement groups in a pool named **data**:

```
# rados list-inconsistent-pg data --format=json-pretty
[0.6]
```

```
rados list-inconsistent-obj PLACEMENT_GROUP_ID
```

For example, list inconsistent objects in a placement group with ID **0.6**:

```
# rados list-inconsistent-obj 0.6
{
    "epoch": 14,
    "inconsistents": [
        {
            "object": {
                "name": "image1",
                "nspace": "",
                "locator": "",
                "snap": "head",
                "version": 1
            },
            "errors": [
                "data_digest_mismatch",
                "size_mismatch"
            ],
            "union_shard_errors": [
                "data_digest_mismatch_oi",
                "size_mismatch_oi"
            ],
            "selected_object_info": "0:602f83fe:::foo:head(16'1 client.4110.0:1
dirty|data_digest|omap_digest s 968 uv 1 dd e978e67f od ffffffff alloc_hint [0 0 0])",
            "shards": [
                {
                    "osd": 0,
                    "errors": [],
                    "size": 968,
                    "omap_digest": "0xffffffff",
                    "data_digest": "0xe978e67f"
                },
                {
                    "osd": 1,
                    "errors": [],
                    "size": 968,
                    "omap_digest": "0xffffffff",
                    "data_digest": "0xe978e67f"
                },
                {
                    "osd": 2,
                    "errors": [
                        "data_digest_mismatch_oi",
```

```
                "size_mismatch_oi"
            ],
            "size": 0,
            "omap_digest": "0xffffffff",
            "data_digest": "0xffffffff"
        }
    ]
  }
]
}
```

The following fields are important to determine what causes the inconsistency:

- **name**: The name of the object with inconsistent replicas.
- **nspace**: The namespace that is a logical separation of a pool. It's empty by default.
- **locator**: The key that is used as the alternative of the object name for placement.
- **snap**: The snapshot ID of the object. The only writable version of the object is called **head**. If an object is a clone, this field includes its sequential ID.
- **version**: The version ID of the object with inconsistent replicas. Each write operation to an object increments it.
- **errors**: A list of errors that indicate inconsistencies between shards without determining which shard or shards are incorrect. See the **shard** array to further investigate the errors.
  - **data_digest_mismatch**: The digest of the replica read from one OSD is different from the other OSDs.
  - **size_mismatch**: The size of a clone or the **head** object does not match the expectation.
  - **read_error**: This error indicates inconsistencies caused most likely by disk errors.
- **union_shard_error**: The union of all errors specific to shards. These errors are connected to a faulty shard. The errors that end with **oi** indicate that you have to compare the information from a faulty object to information with selected objects. See the **shard** array to further investigate the errors.
  In the above example, the object replica stored on **osd.2** has different digest than the replicas stored on **osd.0** and **osd.1**. Specifically, the digest of the replica is not **0xffffffff** as calculated from the shard read from **osd.2**, but **0xe978e67f**. In addition, the size of the replica read from **osd.2** is 0, while the size reported by **osd.0** and **osd.1** is 968.

```
rados list-inconsistent-snapset PLACEMENT_GROUP_ID
```

For example, list inconsistent sets of snapshots (**snapsets**) in a placement group with ID **0.23**:

```
# rados list-inconsistent-snapset 0.23 --format=json-pretty
{
    "epoch": 64,
    "inconsistents": [
        {
            "name": "obj5",
            "nspace": "",
            "locator": "",
```

```
        "snap": "0x00000001",
        "headless": true
    },
    {
        "name": "obj5",
        "nspace": "",
        "locator": "",
        "snap": "0x00000002",
        "headless": true
    },
    {
        "name": "obj5",
        "nspace": "",
        "locator": "",
        "snap": "head",
        "ss_attr_missing": true,
        "extra_clones": true,
        "extra clones": [
            2,
            1
        ]
    }
]
```

The command returns the following errors:

- **ss_attr_missing**: One or more attributes are missing. Attributes are information about snapshots encoded into a snapshot set as a list of key-value pairs.
- **ss_attr_corrupted**: One or more attributes fail to decode.
- **clone_missing**: A clone is missing.
- **snapset_mismatch**: The snapshot set is inconsistent by itself.
- **head_mismatch**: The snapshot set indicates that **head** exists or not, but the scrub results report otherwise.
- **headless**: The **head** of the snapshot set is missing.
- **size_mismatch**: The size of a clone or the **head** object does not match the expectation.

Additional Resources

- *Inconsistent placement groups* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- *Repairing inconsistent placement groups* section in the *Red Hat Ceph Storage Troubleshooting Guide*.

9.5. REPAIRING INCONSISTENT PLACEMENT GROUPS

Due to an error during deep scrubbing, some placement groups can include inconsistencies. Ceph reports such placement groups as **inconsistent**:

```
HEALTH_ERR 1 pgs inconsistent; 2 scrub errors
pg 0.6 is active+clean+inconsistent, acting [0,1,2]
2 scrub errors
```

WARNING

You can repair only certain inconsistencies.

Do not repair the placement groups if the Ceph logs include the following errors:

```
_PG_._ID_ shard _OSD_: soid _OBJECT_ digest _DIGEST_ != known digest _DIGEST_
_PG_._ID_ shard _OSD_: soid _OBJECT_ omap_digest _DIGEST_ != known omap_digest
_DIGEST_
```

Open a support ticket instead. See *Contacting Red Hat Support for service* for details.

Prerequisites

- Root-level access to the Ceph Monitor node.

Procedure

1. Repair the **inconsistent** placement groups:

```
[root@mon ~]# ceph pg repair ID
```

1. Replace ***ID*** with the ID of the **inconsistent** placement group.

Additional Resources

- *Inconsistent placement groups* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- *Listing placement group inconsistencies* *Red Hat Ceph Storage Troubleshooting Guide*.

9.6. INCREASING THE PLACEMENT GROUP

Insufficient Placement Group (PG) count impacts the performance of the Ceph cluster and data distribution. It is one of the main causes of the **nearfull osds** error messages.

The recommended ratio is between 100 and 300 PGs per OSD. This ratio can decrease when you add more OSDs to the cluster.

The **pg_num** and **pgp_num** parameters determine the PG count. These parameters are configured per each pool, and therefore, you must adjust each pool with low PG count separately.

IMPORTANT

Increasing the PG count is the most intensive process that you can perform on a Ceph cluster. This process might have serious performance impact if not done in a slow and methodical way. Once you increase **pgp_num**, you will not be able to stop or reverse the process and you must complete it. Consider increasing the PG count outside of business critical processing time allocation, and alert all clients about the potential performance impact. Do not change the PG count if the cluster is in the **HEALTH_ERR** state.

Prerequisites

- A running Red Hat Ceph Storage cluster in a healthy state.
- Root-level access to the node.

Procedure

1. Reduce the impact of data redistribution and recovery on individual OSDs and OSD hosts:
   a. Lower the value of the **osd max backfills**, **osd_recovery_max_active**, and **osd_recovery_op_priority** parameters:

   ```
   [root@mon ~]# ceph tell osd.* injectargs '--osd_max_backfills 1 --
   osd_recovery_max_active 1 --osd_recovery_op_priority 1'
   ```

   b. Disable the shallow and deep scrubbing:

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

2. Use the *Ceph Placement Groups (PGs) per Pool Calculator* to calculate the optimal value of the **pg_num** and **pgp_num** parameters.
3. Increase the **pg_num** value in small increments until you reach the desired value.
   a. Determine the starting increment value. Use a very low value that is a power of two, and increase it when you determine the impact on the cluster. The optimal value depends on the pool size, OSD count, and client I/O load.
   b. Increment the **pg_num** value:

   ```
   ceph osd pool set POOL pg_num VALUE
   ```

   Specify the pool name and the new value, for example:

   ```
   # ceph osd pool set data pg_num 4
   ```

   c. Monitor the status of the cluster:

   ```
   # ceph -s
   ```

   The PGs state will change from **creating** to **active+clean**. Wait until all PGs are in the **active+clean** state.
4. Increase the **pgp_num** value in small increments until you reach the desired value:

a. Determine the starting increment value. Use a very low value that is a power of two, and increase it when you determine the impact on the cluster. The optimal value depends on the pool size, OSD count, and client I/O load.

b. Increment the **pgp_num** value:

```
ceph osd pool set POOL pgp_num VALUE
```

Specify the pool name and the new value, for example:

```
# ceph osd pool set data pgp_num 4
```

c. Monitor the status of the cluster:

```
# ceph -s
```

The PGs state will change through **peering**, **wait_backfill**, **backfilling**, **recover**, and others. Wait until all PGs are in the **active+clean** state.

5. Repeat the previous steps for all pools with insufficient PG count.

6. Set **osd max backfills**, **osd_recovery_max_active**, and **osd_recovery_op_priority** to their default values:

```
# ceph tell osd.* injectargs '--osd_max_backfills 1 --osd_recovery_max_active 3 --
osd_recovery_op_priority 3'
```

7. Enable the shallow and deep scrubbing:

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

Additional Resources

- *Nearfull OSDs*
- The *Monitoring Placement Group Sets* section in the *Administration Guide* for Red Hat Ceph Storage 4

9.7. ADDITIONAL RESOURCES

- See Chapter 3, *Troubleshooting networking issues* for details.
- See Chapter 4, *Troubleshooting Ceph Monitors* for details about troubleshooting the most common errors related to Ceph Monitors.
- See Chapter 5, *Troubleshooting Ceph OSDs* for details about troubleshooting the most common errors related to Ceph OSDs.

CHAPTER 10. TROUBLESHOOTING CEPH OBJECTS

As a storage administrator, you can use the **ceph-objectstore-tool** utility to perform high-level or low-level object operations. The **ceph-objectstore-tool** utility can help you troubleshoot problems related to objects within a particular OSD or placement group.

You can also start OSD containers in rescue/maintenance mode to repair OSDs without installing Ceph packages on the OSD node.

IMPORTANT

Manipulating objects can cause unrecoverable data loss. Contact Red Hat support before using the **ceph-objectstore-tool** utility.

10.1. PREREQUISITES

- Verify there are no network-related issues.

10.2. TROUBLESHOOTING CEPH OBJECTS IN A CONTAINERIZED ENVIRONMENT

The OSD container can be started in rescue/maintenance mode to repair OSDs in Red Hat Ceph Storage 4 without installing Ceph packages on the OSD node.

You can use **ceph-bluestore-tool** to run consistency check with **fsck** command, or to run consistency check and repair any errors with **repair** command.

IMPORTANT

This procedure is specific to containerized deployments only. Skip this section for bare-metal deployments

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Set **noout** flag on cluster.

   Example

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

2. Login to the node hosting the OSD container.
3. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

   Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
```

4. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

   Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

5. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

   Example

```
# Please do not change this file directly since it is managed by Ansible and will be overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid
```

```
[Install]
WantedBy=multi-user.target
```

6. Reload **systemd** manager configuration.

   Example

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

7. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   systemctl restart ceph-osd@OSD_ID.service
   ```

   Replace ***OSD_ID*** with the ID of the OSD.

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

8. Login to the container associated with the ***OSD_ID***.

   Syntax

   ```
   podman exec -it ceph-osd-OSD_ID /bin/bash
   ```

   Example

   ```
   [root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
   ```

9. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

   Syntax

   ```
   ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
   ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
   ```

   Example

   ```
   [root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
           osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
   [root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
   9583ba64a4ff
   Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
   Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
   Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
   /dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-
   83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
   Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-
   data-d073e8b3-0b89-4271-af5b-83045fd000dc /var/lib/ceph/osd/ceph-0/block
   ```

```
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--4685--
9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-40a3-
8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-0-
087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service →
/usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

10. Run **fsck** and **repair** commands.

    Syntax

    ```
    ceph-bluestore-tool fsck --path /var/lib/ceph/osd/ceph-OSD_ID
    ceph-bluestore-tool repair --path /var/lib/ceph/osd/ceph-OSD_ID
    ```

    Example

    ```
    [root@osd ~]# ceph-bluestore-tool fsck --path /var/lib/ceph/osd/ceph-0
    fsck success
    ```

    ```
    [root@osd ~]# ceph-bluestore-tool repair --path /var/lib/ceph/osd/ceph-0
    repair success
    ```

11. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

    Example

    ```
    [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.modified
    [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-osd@.service
    ```

12. Reload **systemd** manager configuration.

    Example

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

13. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

    Example

    ```
    [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
    ```

14. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
[root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
```

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.3. TROUBLESHOOTING HIGH-LEVEL OBJECT OPERATIONS

As a storage administrator, you can use the **ceph-objectstore-tool** utility to perform high-level object operations. The **ceph-objectstore-tool** utility supports the following high-level object operations:

- List objects
- List lost objects
- Fix lost objects

IMPORTANT

Manipulating objects can cause unrecoverable data loss. Contact Red Hat support before using the **ceph-objectstore-tool** utility.

10.3.1. Prerequisites

- Root-level access to the Ceph OSD nodes.

10.3.2. Listing objects

The OSD can contain zero to many placement groups, and zero to many objects within a placement group (PG). The **ceph-objectstore-tool** utility allows you to list objects stored within an OSD.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@OSD_NUMBER
```

Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

      Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

      Example

      ```
      [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
      ```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

      Example

      ```
      [root@osd ~]# mv /run/ceph-osd@0.service-cid /root
      ```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

      Example

      ```
      # Please do not change this file directly since it is managed by Ansible and will be overwritten
      [Unit]
      Description=Ceph OSD
      After=network.target

      [Service]
      EnvironmentFile=-/etc/environment
      ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
      ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
      ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
        -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
        --rm \
        --net=host \
        --privileged=true \
        --pid=host \
        --ipc=host \
        --cpus=2 \
        -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. Identify all the objects within an OSD, regardless of their placement group:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --op list
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list
```

4. Identify all the objects within a placement group:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID --op list
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c --op list
```

5. Identify the PG an object belongs to:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --op list OBJECT_ID
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list
default.region
```

6. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.3.3. Listing lost objects

An OSD can mark objects as *lost* or *unfound*. You can use the **ceph-objectstore-tool** to list the *lost and unfound* objects stored within an OSD.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@OSD_NUMBER
```

Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

   Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
   ```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /root
   ```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

   Example

   ```
   # Please do not change this file directly since it is managed by Ansible and will be overwritten
   [Unit]
   Description=Ceph OSD
   After=network.target

   [Service]
   EnvironmentFile=-/etc/environment
   ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
   ```

```
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
        osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. Use the **ceph-objectstore-tool** utility to list *lost and unfound* objects. Select the appropriate circumstance:

   a. To list all the lost objects:

   -

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --op list-lost
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list-lost
```

b. To list all the lost objects within a placement group:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID --op list-lost
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c --op list-lost
```

c. To list a lost object by its identifier:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --op list-lost OBJECT_ID
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list-lost default.region
```

4. For containerized deployments, to revert the changes, follow the below steps:

a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.modified
[root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-osd@.service
```

b. Reload **systemd** manager configuration.

Example

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

c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
```

d. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
[root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
```

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting a Ceph Daemons by Instance* section in the Red Hat Ceph Storage Administration Guide.

10.3.4. Fixing lost objects

You can use the **ceph-objectstore-tool** utility to list and fix *lost and unfound* objects stored within a Ceph OSD. This procedure applies only to legacy objects.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

   Syntax

   ```
   [root@osd ~]# systemctl status ceph-osd@OSD_NUMBER
   ```

   Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

      Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

      Example

      ```
      [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
      ```

d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

Example

```
# Please do not change this file directly since it is managed by Ansible and will be
overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid
```

```
[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
        osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
```

```
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. To list all the lost legacy objects:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost --dry-run
   ```

   Example

   ```
   [root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op fix-lost --dry-
   run
   ```

4. Use the **ceph-objectstore-tool** utility to fix *lost and unfound* objects as a **ceph** user. Select the appropriate circumstance:

   - To fix all lost objects:

     Syntax

     ```
     su - ceph -c 'ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost'
     ```

     Example

     ```
     [root@osd ~]# su - ceph -c 'ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --
     op fix-lost'
     ```

   - To fix all the lost objects within a placement group:

     ```
     su - ceph -c 'ceph-objectstore-tool --data-path _PATH_TO_OSD_ --pgid _PG_ID_ --op
     fix-lost'
     ```

     Example

     ```
     [root@osd ~]# su - ceph -c 'ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --
     pgid 0.1c --op fix-lost'
     ```

   - To fix a lost object by its identifier:

Syntax

```
su - ceph -c 'ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost OBJECT_ID'
```

Example

```
[root@osd ~]# su - ceph -c 'ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --
op fix-lost default.region'
```

5. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

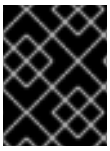
Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4. TROUBLESHOOTING LOW-LEVEL OBJECT OPERATIONS

As a storage administrator, you can use the **ceph-objectstore-tool** utility to perform low-level object operations. The **ceph-objectstore-tool** utility supports the following low-level object operations:

- Manipulate the object's content
- Remove an object
- List the object map (OMAP)
- Manipulate the OMAP header
- Manipulate the OMAP key
- List the object's attributes
- Manipulate the object's attribute key



IMPORTANT

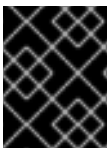
Manipulating objects can cause unrecoverable data loss. Contact Red Hat support before using the **ceph-objectstore-tool** utility.

10.4.1. Prerequisites

- Root-level access to the Ceph OSD nodes.

10.4.2. Manipulating the object's content

With the **ceph-objectstore-tool** utility, you can get or set bytes on an object.



IMPORTANT

Setting the bytes on an object can cause unrecoverable data loss. To prevent data loss, make a backup copy of the object.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@$OSD_NUMBER
```

   Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

Example

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

b. Login to the node hosting the OSD container.

c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
```

d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

Example

```
# Please do not change this file directly since it is managed by Ansible and will be overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
```

```
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. Find the object by listing the objects of the OSD or placement group (PG).

4. Before setting the bytes on an object, make a backup and a working copy of the object:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
get-bytes > OBJECT_FILE_NAME

[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
get-bytes > OBJECT_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c \
'{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
get-bytes > zone_info.default.backup

[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c \
'{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
get-bytes > zone_info.default.working-copy
```

5. Edit the working copy object file and modify the object contents accordingly.

6. Set the bytes of the object:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
set-bytes < OBJECT_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c \
'{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
set-bytes < zone_info.default.working-copy
```

7. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.3. Removing an object

Use the **ceph-objectstore-tool** utility to remove an object. By removing an object, its contents and references are removed from the placement group (PG).

IMPORTANT

You cannot recreate an object once it is removed.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@$OSD_NUMBER
```

Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:
   a. Set **noout** flag on cluster.

Example

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

   b. Login to the node hosting the OSD container.
   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

Example

```
# Please do not change this file directly since it is managed by Ansible and will be
overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
```

```
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. Remove an object:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
   OBJECT \
   remove
   ```

   Example

   ```
   [root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c \
   '{"oid":"zone_info.default","key":"","snapid":-
   2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
   remove
   ```

4. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

      Example

      ```
      [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
      osd@.service.modified
      [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
      osd@.service
      ```

   b. Reload **systemd** manager configuration.

      Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

      Example

      ```
      [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
      ```

   d. Restart the OSD service associated with the ***OSD_ID***.

      Syntax

      ```
      [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
      ```

      Example

      -

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.4. Listing the object map

Use the **ceph-objectstore-tool** utility to list the contents of the object map (OMAP). The output provides you a list of keys.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

   ```
   [root@osd ~]# systemctl status ceph-osd@OSD_NUMBER
   ```

   Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

      Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

      Example

      ```
      [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
      ```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

      Example

      ```
      [root@osd ~]# mv /run/ceph-osd@0.service-cid /root
      ```

e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

Example

```
# Please do not change this file directly since it is managed by Ansible and will be
overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
```

```
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. List the object map:

```
[root@osd ~]# ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
list-omap
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c \
'{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
list-omap
```

4. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.5. Manipulating the object map header

The **ceph-objectstore-tool** utility will output the object map (OMAP) header with the values associated with the object's keys.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. For containerized deployments, to access the bluestore tool, follow the below steps:
   a. Set **noout** flag on cluster.

      Example

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

   b. Login to the node hosting the OSD container.
   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

      Example

      ```
      [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
      ```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

      Example

      ```
      [root@osd ~]# mv /run/ceph-osd@0.service-cid /root
      ```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

      Example

      ```
      # Please do not change this file directly since it is managed by Ansible and will be overwritten
      [Unit]
      Description=Ceph OSD
      After=network.target
      ```

```
[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

2. Verify the appropriate OSD is down:

Syntax

```
systemctl status ceph-osd@OSD_NUMBER
```

Example

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

3. Get the object map header:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
get-omaphdr > OBJECT_MAP_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
get-omaphdr > zone_info.default.omaphdr.txt
```

4. Set the object map header:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
get-omaphdr < OBJECT_MAP_FILE_NAME
```

Example

```
[root@osd ~]# su - ceph -c 'ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
set-omaphdr < zone_info.default.omaphdr.txt
```

5. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

Example

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

c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
```

d. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
[root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
```

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.6. Manipulating the object map key

Use the **ceph-objectstore-tool** utility to change the object map (OMAP) key. You need to provide the data path, the placement group identifier (PG ID), the object, and the key in the OMAP.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

   Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

   Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
osd@.service.backup
```

d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

Example

```
# Please do not change this file directly since it is managed by Ansible and will be
overwritten
[Unit]
Description=Ceph OSD
After=network.target

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
```

```
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
        osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
```

```
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

2. Get the object map key:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
get-omap KEY > OBJECT_MAP_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
get-omap "" > zone_info.default.omap.txt
```

3. Set the object map key:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
set-omap KEY < OBJECT_MAP_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
set-omap "" < zone_info.default.omap.txt
```

- Remove the object map key:

  Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
rm-omap KEY
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}'  \
rm-omap ""
```

4. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.7. Listing the object's attributes

Use the **ceph-objectstore-tool** utility to list an object's attributes. The output provides you with the object's keys and values.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@OSD_NUMBER
```

Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:
   a. Set **noout** flag on cluster.

   Example

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

   b. Login to the node hosting the OSD container.
   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
   ```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /root
   ```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

   Example

   ```
   # Please do not change this file directly since it is managed by Ansible and will be overwritten
   [Unit]
   Description=Ceph OSD
   After=network.target
   ```

```
[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. List the object's attributes:

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
list-attrs
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
list-attrs
```

4. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   [root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-
   osd@.service
   ```

   b. Reload **systemd** manager configuration.

   Example

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

   c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

   Example

   ```
   [root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
   ```

   d. Restart the OSD service associated with the ***OSD_ID***.

   Syntax

   ```
   [root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
   ```

   Example

   ```
   [root@osd ~]# systemctl restart ceph-osd@0.service
   ```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.4.8. Manipulating the object attribute key

Use the **ceph-objectstore-tool** utility to change an object's attributes. To manipulate the object's attributes you need the data and journal paths, the placement group identifier (PG ID), the object, and the key in the object's attribute.

Prerequisites

- Root-level access to the Ceph OSD node.
- Stopping the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

```
[root@osd ~]# systemctl status ceph-osd@$OSD_NUMBER
```

   Example

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

2. For containerized deployments, to access the bluestore tool, follow the below steps:

   a. Set **noout** flag on cluster.

   Example

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

   b. Login to the node hosting the OSD container.

   c. Backup **/etc/systemd/system/ceph-osd@.service** unit file to **/root** directory.

   Example

```
[root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-osd@.service.backup
```

   d. Move **/run/ceph-osd@OSD_ID.service-cid** file to **/root**.

   Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /root
```

   e. Edit **/etc/systemd/system/ceph-osd@.service** unit file and add **-it --entrypoint /bin/bash** option to podman command.

   Example

```
# Please do not change this file directly since it is managed by Ansible and will be overwritten
[Unit]
Description=Ceph OSD
After=network.target
```

```
[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/rm -f /%t/%n-pid /%t/%n-cid
ExecStartPre=-/usr/bin/podman rm -f ceph-osd-%i
ExecStart=/usr/bin/podman run -it --entrypoint /bin/bash \
  -d --conmon-pidfile /%t/%n-pid --cidfile /%t/%n-cid \
  --rm \
  --net=host \
  --privileged=true \
  --pid=host \
  --ipc=host \
  --cpus=2 \
  -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 \
  -e OSD_BLUESTORE=1 -e OSD_FILESTORE=0 -e OSD_DMCRYPT=0 \
  -e CLUSTER=ceph \
  -v /run/lvm/:/run/lvm/ \
  -e CEPH_DAEMON=OSD_CEPH_VOLUME_ACTIVATE \
  -e CONTAINER_IMAGE=registry.redhat.io/rhceph/rhceph-4-rhel8:latest \
  -e OSD_ID=%i \
  -e DEBUG=stayalive \
  --name=ceph-osd-%i \
   \
  registry.redhat.io/rhceph/rhceph-4-rhel8:latest
ExecStop=-/usr/bin/sh -c "/usr/bin/podman rm -f `cat /%t/%n-cid`"
KillMode=none
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15
Type=forking
PIDFile=/%t/%n-pid

[Install]
WantedBy=multi-user.target
```

f. Reload **systemd** manager configuration.

Example

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

g. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
systemctl restart ceph-osd@OSD_ID.service
```

Replace ***OSD_ID*** with the ID of the OSD.

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

h. Login to the container associated with the ***OSD_ID***.

Syntax

```
podman exec -it ceph-osd-OSD_ID /bin/bash
```

Example

```
[root@osd ~]# podman exec -it ceph-osd-0 /bin/bash
```

i. Get **osd fsid** and activate the OSD to mount OSD's logical volume (LV).

Syntax

```
ceph-volume lvm list |grep -A15 "osd\.OSD_ID"|grep "osd fsid"
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

Example

```
[root@osd ~]# ceph-volume lvm list |grep -A15 "osd\.0"|grep "osd fsid"
            osd fsid                  087eee15-6561-40a3-8fe4-9583ba64a4ff
[root@osd ~]# ceph-volume lvm activate --bluestore 0 087eee15-6561-40a3-8fe4-
9583ba64a4ff
Running command: /usr/bin/mount -t tmpfs tmpfs /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/ceph-bluestore-tool --cluster=ceph prime-osd-dir --dev
/dev/ceph-41c69f8f-30e2-4685-9c5c-c605898c5537/osd-data-d073e8b3-0b89-4271-
af5b-83045fd000dc --path /var/lib/ceph/osd/ceph-0 --no-mon-config
Running command: /usr/bin/ln -snf /dev/ceph-41c69f8f-30e2-4685-9c5c-
c605898c5537/osd-data-d073e8b3-0b89-4271-af5b-83045fd000dc
/var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -h ceph:ceph /var/lib/ceph/osd/ceph-0/block
Running command: /usr/bin/chown -R ceph:ceph /dev/mapper/ceph--41c69f8f--30e2--
4685--9c5c--c605898c5537-osd--data--d073e8b3--0b89--4271--af5b--83045fd000dc
Running command: /usr/bin/chown -R ceph:ceph /var/lib/ceph/osd/ceph-0
Running command: /usr/bin/systemctl enable ceph-volume@lvm-0-087eee15-6561-
40a3-8fe4-9583ba64a4ff
 stderr: Created symlink /etc/systemd/system/multi-user.target.wants/ceph-volume@lvm-
0-087eee15-6561-40a3-8fe4-9583ba64a4ff.service → /usr/lib/systemd/system/ceph-
volume@.service.
Running command: /usr/bin/systemctl enable --runtime ceph-osd@0
 stderr: Created symlink /run/systemd/system/ceph-osd.target.wants/ceph osd@0.service
→ /usr/lib/systemd/system/ceph-osd@.service.
Running command: /usr/bin/systemctl start ceph-osd@0
 stderr: Running in chroot, ignoring request: start
--> ceph-volume lvm activate successful for osd ID: 0
```

3. Get the object's attributes:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
get-attrs KEY > OBJECT_ATTRS_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
get-attrs "oid" > zone_info.default.attr.txt
```

4. Set an object's attributes:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
set-attrs KEY < OBJECT_ATTRS_FILE_NAME
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
set-attrs "oid" < zone_info.default.attr.txt
```

5. Remove an object's attributes:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD \
--pgid PG_ID OBJECT \
rm-attrs KEY
```

Example

```
[root@osd ~]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 \
--pgid 0.1c '{"oid":"zone_info.default","key":"","snapid":-
2,"hash":235010478,"max":0,"pool":11,"namespace":""}' \
rm-attrs "oid"
```

6. For containerized deployments, to revert the changes, follow the below steps:

   a. After exiting the container, copy **/etc/systemd/system/ceph-osd@.service** unit file from **/root** directory.

   Example

   ```
   [root@osd ~]# cp /etc/systemd/system/ceph-osd@.service /root/ceph-
   osd@.service.modified
   ```

```
[root@osd ~]# cp /root/ceph-osd@.service.backup /etc/systemd/system/ceph-osd@.service
```

b. Reload **systemd** manager configuration.

Example

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

c. Move **/run/ceph-osd@*OSD_ID*.service-cid** file to **/tmp**.

Example

```
[root@osd ~]# mv /run/ceph-osd@0.service-cid /tmp
```

d. Restart the OSD service associated with the ***OSD_ID***.

Syntax

```
[root@osd ~]# systemctl restart ceph-osd@OSD_ID.service
```

Example

```
[root@osd ~]# systemctl restart ceph-osd@0.service
```

Additional Resources

- For more information on stopping an OSD, see the *Starting, Stopping, and Restarting the Ceph Daemons by Instance* section in the *Red Hat Ceph Storage Administration Guide*.

10.5. ADDITIONAL RESOURCES

- For Red Hat Ceph Storage support, see the Red Hat Customer Portal.

CHAPTER 11. CONTACTING RED HAT SUPPORT FOR SERVICE

If the information in this guide did not help you to solve the problem, this chapter explains how you contact the Red Hat support service.

11.1. PREREQUISITES

- Red Hat support account.

11.2. PROVIDING INFORMATION TO RED HAT SUPPORT ENGINEERS

If you are unable to fix problems related to Red Hat Ceph Storage, contact the Red Hat Support Service and provide sufficient amount of information that helps the support engineers to faster troubleshoot the problem you encounter.

Prerequisites

- Root-level access to the node.
- Red Hat support account.

Procedure

1. Open a support ticket on the Red Hat Customer Portal .
2. Ideally, attach an **sosreport** to the ticket. See the What is a sosreport and how to create one in Red Hat Enterprise Linux 4.6 and later? solution for details.
3. If the Ceph daemons fail with a segmentation fault, consider generating a human-readable core dump file. See Generating readable core dump files for details.

11.3. GENERATING READABLE CORE DUMP FILES

When a Ceph daemon terminates unexpectedly with a segmentation fault, gather the information about its failure and provide it to the Red Hat Support Engineers.

Such information speeds up the initial investigation. Also, the Support Engineers can compare the information from the core dump files with Red Hat Ceph Storage cluster known issues.

11.3.1. Prerequisites

1. Install the **ceph-debuginfo** package if it is not installed already.
   a. Enable the repository containing the **ceph-debuginfo** package:

   Red Hat Enterprise Linux 7:

   ```
   subscription-manager repos --enable=rhel-7-server-rhceph-4-DAEMON-debug-rpms
   ```

   Replace ***DAEMON*** with **osd** or **mon** depending on the type of Ceph node.

   Red Hat Enterprise Linux 8:

```
subscription-manager repos --enable=rhceph-4-tools-for-rhel-8-x86_64-debug-rpms
```

b. Install the **ceph-debuginfo** package:

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

2. Ensure that the **gdb** package is installed and if it is not, install it:

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

Continue with the procedure based on the type of your deployment:

- Section 11.3.2, "Generating readable core dump files on bare-metal deployments"
- Section 11.3.3, "Generating readable core dump files in containerized deployments"

11.3.2. Generating readable core dump files on bare-metal deployments

Follow this procedure to generate a core dump file if you use Red Hat Ceph Storage on bare-metal.

Procedure

1. Enable generating core dump files for Ceph.

a. Set the proper **ulimits** for the core dump files by adding the following parameter to the **/etc/systemd/system.conf** file:

```
DefaultLimitCORE=infinity
```

b. Comment out the **PrivateTmp=true** parameter in the Ceph daemon service file, by default located at **/lib/systemd/system/*CLUSTER_NAME-DAEMON*@.service**:

```
[root@mon ~]# PrivateTmp=true
```

c. Set the **suid_dumpable** flag to **2** to allow the Ceph daemons to generate dump core files:

```
[root@mon ~]# sysctl fs.suid_dumpable=2
```

d. Adjust the core dump files location:

```
[root@mon ~]# sysctl kernel.core_pattern=/tmp/core
```

e. Modify **/etc/systemd/coredump.conf** file by adding the following lines under section **[Coredump]**:

```
ProcessSizeMax=8G
ExternalSizeMax=8G
JournalSizeMax=8G
```

f. Reload the **systemd** service for the changes to take effect:

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

g. Restart the Ceph daemon for the changes to take effect:

```
[root@mon ~]# systemctl restart ceph-DAEMON@ID
```

Specify the daemon type (**osd** or **mon**) and its ID (numbers for OSDs, or short host names for Monitors) for example:

```
[root@mon ~]# systemctl restart ceph-osd@1
```

2. Reproduce the failure, for example try to start the daemon again.

3. Use the GNU Debugger (GDB) to generate a readable backtrace from an application core dump file:

```
gdb /usr/bin/ceph-DAEMON /tmp/core.PID
```

Specify the daemon type and the PID of the failed process, for example:

```
$ gdb /usr/bin/ceph-osd /tmp/core.123456
```

In the GDB command prompt disable paging and enable logging to a file by entering the commands **set pag off** and **set log on**:

```
(gdb) set pag off
(gdb) set log on
```

Apply the **backtrace** command to all threads of the process by entering **thr a a bt full**:

```
(gdb) thr a a bt full
```

After the backtrace is generated turn off logging by entering **set log off**:

```
(gdb) set log off
```

4. Transfer the log file **gdb.txt** to the system you access the Red Hat Customer Portal from and attach it to a support ticket.

11.3.3. Generating readable core dump files in containerized deployments

Follow this procedure to generate a core dump file if you use Red Hat Ceph Storage in containers. The procedure involves two scenarios of capturing the core dump file:

- When a Ceph process terminates unexpectedly due to the SIGILL, SIGTRAP, SIGABRT, or SIGSEGV error.

or

- Manually, for example for debugging issues such as Ceph processes are consuming high CPU cycles, or are not responding.

Prerequisites

- Root-level access to the container node running the Ceph containers.

- Installation of the appropriate debugging packages.
- Installation of the GNU Project Debugger (**gdb**) package.

Procedure

1. If a Ceph process terminates unexpectedly due to the SIGILL, SIGTRAP, SIGABRT, or SIGSEGV error:

   a. Set the core pattern to the **systemd-coredump** service on the node where the container with the failed Ceph process is running, for example:

   ```
   [root@mon]# echo "| /usr/lib/systemd/systemd-coredump %P %u %g %s %t %e" >
   /proc/sys/kernel/core_pattern
   ```

   b. Watch for the next container failure due to a Ceph process and search for a core dump file in the **/var/lib/systemd/coredump/** directory, for example:

   ```
   [root@mon]# ls -ltr /var/lib/systemd/coredump
   total 8232
   -rw-r-----. 1 root root 8427548 Jan 22 19:24 core.ceph-
   osd.167.5ede29340b6c4fe4845147f847514c12.15622.1584573794000000.xz
   ```

2. To manually capture a core dump file for the **Ceph Monitors** and **Ceph Managers**:

   a. Get the **ceph-mon** package details of the Ceph daemon from the container:

   Red Hat Enterprise Linux 7:

   ```
   [root@mon]# docker exec -it NAME /bin/bash
   [root@mon]# rpm -qa | grep ceph
   ```

   Red Hat Enterprise Linux 8:

   ```
   [root@mon]# podman exec -it NAME /bin/bash
   [root@mon]# rpm -qa | grep ceph
   ```

   Replace *NAME* with the name of the Ceph container.

   b. Make a backup copy and open for editing the **ceph-mon@.service** file:

   ```
   [root@mon]# cp /etc/systemd/system/ceph-mon@.service /etc/systemd/system/ceph-
   mon@.service.orig
   ```

   c. In the **ceph-mon@.service** file, add these three options to the **[Service]** section, each on a separate line:

   ```
   --pid=host \
   --ipc=host \
   --cap-add=SYS_PTRACE \
   ```

   Example

```
[Unit]
Description=Ceph Monitor
After=docker.service

[Service]
EnvironmentFile=-/etc/environment
ExecStartPre=-/usr/bin/docker rm ceph-mon-%i
ExecStartPre=/bin/sh -c '"$(command -v mkdir)" -p /etc/ceph /var/lib/ceph/mon'
ExecStart=/usr/bin/docker run --rm --name ceph-mon-%i \
  --memory=924m \
--cpu-quota=100000 \
-v /var/lib/ceph:/var/lib/ceph:z \
  -v /etc/ceph:/etc/ceph:z \
  -v /var/run/ceph:/var/run/ceph:z \
-v /etc/localtime:/etc/localtime:ro \
--net=host \
--privileged=true \
--ipc=host \ 1
--pid=host \ 2
--cap-add=SYS_PTRACE \ 3
-e IP_VERSION=4 \
       -e MON_IP=10.74.131.17 \
      -e CLUSTER=ceph \
  -e FSID=9448efca-b1a1-45a3-bf7b-b55cba696a6e \
  -e CEPH_PUBLIC_NETWORK=10.74.131.0/24 \
  -e CEPH_DAEMON=MON \
   \
  registry.access.redhat.com/rhceph/rhceph-3-rhel7:latest
ExecStop=-/usr/bin/docker stop ceph-mon-%i
ExecStopPost=-/bin/rm -f /var/run/ceph/ceph-mon.pd-cephcontainer-mon01.asok
Restart=always
RestartSec=10s
TimeoutStartSec=120
TimeoutStopSec=15

[Install]
WantedBy=multi-user.target
```

d. Restart the Ceph Monitor daemon:

Syntax

```
systemctl restart ceph-mon@MONITOR_ID
```

Replace *MONITOR_ID* with the ID number of the Ceph Monitor.

Example

```
[root@mon]# systemctl restart ceph-mon@1
```

e. Install the **gdb** package inside the Ceph Monitor container:

Red Hat Enterprise Linux 7:

```
[root@mon]# docker exec -it ceph-mon-MONITOR_ID /bin/bash
sh $ yum install gdb
```

Red Hat Enterprise Linux 8:

```
[root@mon]# podman exec -it ceph-mon-MONITOR_ID /bin/bash
sh $ yum install gdb
```

Replace *MONITOR_ID* with the ID number of the Ceph Monitor.

f. Find the process ID:

Syntax

```
ps -aef | grep PROCESS | grep -v run
```

Replace *PROCESS* with the name of failed process, for example **ceph-mon**.

Example

```
[root@mon]# ps -aef | grep ceph-mon | grep -v run
ceph       15390   15266  0 18:54 ?        00:00:29 /usr/bin/ceph-mon --cluster ceph --
setroot ceph --setgroup ceph -d -i 5
ceph       18110   17985  1 19:40 ?        00:00:08 /usr/bin/ceph-mon --cluster ceph --
setroot ceph --setgroup ceph -d -i 2
```

g. Generate the core dump file:

Syntax

```
gcore ID
```

Replace *ID* with the ID of the failed process that you got from the previous step, for example **18110**:

Example

```
[root@mon]# gcore 18110
warning: target file /proc/18110/cmdline contained unexpected null characters
Saved corefile core.18110
```

h. Verify that the core dump file has been generated correctly.

Example

```
[root@mon]# ls -ltr
total 709772
-rw-r--r--. 1 root root 726799544 Mar 18 19:46 core.18110
```

i. Copy the core dump file outside of the Ceph Monitor container:

Red Hat Enterprise Linux 7:

```
[root@mon]# docker cp ceph-mon-MONITOR_ID:/tmp/mon.core.MONITOR_PID /tmp
```

Red Hat Enterprise Linux 8:

```
[root@mon]# podman cp ceph-mon-MONITOR_ID:/tmp/mon.core.MONITOR_PID /tmp
```

Replace *MONITOR_ID* with the ID number of the Ceph Monitor and replace *MONITOR_PID* with the process ID number.

j. Restore the backup copy of the **ceph-mon@.service** file:

```
[root@mon]# cp /etc/systemd/system/ceph-mon@.service.orig
/etc/systemd/system/ceph-mon@.service
```

k. Restart the Ceph Monitor daemon:

Syntax

```
systemctl restart ceph-mon@MONITOR_ID
```

Replace *MONITOR_ID* with the ID number of the Ceph Monitor.

Example

```
[root@mon]# systemctl restart ceph-mon@1
```

l. Upload the core dump file for analysis by Red Hat support, see step 4.

3. To manually capture a core dump file for **Ceph OSDs**:

a. Get the **ceph-osd** package details of the Ceph daemon from the container:

Red Hat Enterprise Linux 7:

```
[root@osd]# docker exec -it NAME /bin/bash
[root@osd]# rpm -qa | grep ceph
```

Red Hat Enterprise Linux 8:

```
[root@osd]# podman exec -it NAME /bin/bash
[root@osd]# rpm -qa | grep ceph
```

Replace *NAME* with the name of the Ceph container.

b. Install the Ceph package for the same version of the **ceph-osd** package on the node where the Ceph containers are running:

Red Hat Enterprise Linux 7:

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

Red Hat Enterprise Linux 8:

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

If needed, enable the appropriate repository first. See the *Enabling the Red Hat Ceph Storage repositories* section in the Installation Guide for details.

c. Find the ID of the process that has failed:

```
ps -aef | grep PROCESS | grep -v run
```

Replace *PROCESS* with the name of failed process, for example **ceph-osd**.

```
[root@osd]# ps -aef | grep ceph-osd | grep -v run
ceph       15390   15266  0 18:54 ?        00:00:29 /usr/bin/ceph-osd --cluster ceph --
setroot ceph --setgroup ceph -d -i 5
ceph       18110   17985  1 19:40 ?        00:00:08 /usr/bin/ceph-osd --cluster ceph --
setroot ceph --setgroup ceph -d -i 2
```

d. Generate the core dump file:

```
gcore ID
```

Replace *ID* with the ID of the failed process that you got from the previous step, for example **18110**:

```
[root@osd]# gcore 18110
warning: target file /proc/18110/cmdline contained unexpected null characters
Saved corefile core.18110
```

e. Verify that the core dump file has been generated correctly.

```
[root@osd]# ls -ltr
total 709772
-rw-r--r--. 1 root root 726799544 Mar 18 19:46 core.18110
```

f. Upload the core dump file for analysis by Red Hat support, see the next step.

4. Upload the core dump file for analysis to a Red Hat support case. See *Providing information to Red Hat Support engineers* for details.

11.3.4. Additional Resources

- The How to use gdb to generate a readable backtrace from an application core solution on the Red Hat Customer Portal
- The How to enable core file dumps when an application crashes or segmentation faults solution on the Red Hat Customer Portal

APPENDIX A. CEPH SUBSYSTEMS DEFAULT LOGGING LEVEL VALUES

A table of the default logging level values for the various Ceph subsystems.

Subsystem	Log Level	Memory Level
asok	1	5
auth	1	5
buffer	0	0
client	0	5
context	0	5
crush	1	5
default	0	5
filer	0	5
bluestore	1	5
finisher	1	5
heartbeatmap	1	5
javaclient	1	5
journaler	0	5
journal	1	5
lockdep	0	5
mds balancer	1	5
mds locker	1	5
mds log expire	1	5
mds log	1	5
mds migrator	1	5

Subsystem	Log Level	Memory Level
mds	1	5
monc	0	5
mon	1	5
ms	0	5
objclass	0	5
objectcacher	0	5
objecter	0	0
optracker	0	5
osd	0	5
paxos	0	5
perfcounter	1	5
rados	0	5
rbd	0	5
rgw	1	5
throttle	1	5
timer	0	5
tp	0	5