Red Hat Ceph Storage 6

Troubleshooting Guide

Troubleshooting Red Hat Ceph Storage

Last Updated: 2024-03-27

Red Hat Ceph Storage 6 Troubleshooting Guide

Troubleshooting Red Hat Ceph Storage

Legal Notice

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

As a storage administrator, you can do the initial troubleshooting of a Red Hat Ceph Storage cluster before contacting Red Hat support. This chapter includes the following information:

- *Identifying problems.*
- *Diagnosing the health of a storage cluster .*
- *Understanding Ceph Health.*
- *Muting health alerts of a Ceph cluster .*
- *Understanding Ceph logs.*
- *Generating an `sos report` .*

Prerequisites

- A running Red Hat Ceph Storage cluster.

1.1. 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. 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. Log into the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Check the overall status of the storage cluster:

   Example

   ```
   [ceph: root@host01 /]# ceph health detail
   ```

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

3. Monitor the logs of the storage cluster:

   Example

   ```
   [ceph: root@host01 /]# ceph -W cephadm
   ```

4. To capture the logs of the cluster to a file, run the following commands:

   Example

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

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

5. If the logs do not include a 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

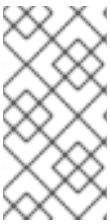
The **ceph health** command returns information about the status of the Red Hat Ceph Storage cluster:

- **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.



NOTE

A health warning is displayed if there is no **mgr** daemon running. In case the last **mgr** daemon of a Red Hat Ceph Storage cluster was removed, you can manually deploy a **mgr** daemon, on a random host of the Red Hat Storage cluster. See the *Manually deploying a mgr daemon* in the *Red Hat Ceph Storage 6 Administration Guide*.

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. MUTING HEALTH ALERTS OF A CEPH CLUSTER

In certain scenarios, users might want to temporarily mute some warnings, because they are already aware of the warning and cannot act on it right away. You can mute health checks so that they do not affect the overall reported status of the Ceph cluster.

Alerts are specified using the health check codes. One example is, when an OSD is brought down for maintenance, **OSD_DOWN** warnings are expected. You can choose to mute the warning until the maintenance is over because those warnings put the cluster in **HEALTH_WARN** instead of **HEALTH_OK** for the entire duration of maintenance.

Most health mutes also disappear if the extent of an alert gets worse. For example, if there is one OSD down, and the alert is muted, the mute disappears if one or more additional OSDs go down. This is true for any health alert that involves a count indicating how much or how many of something is triggering the warning or error.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level of access to the nodes.
- A health warning message.

Procedure

1. Log into the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Check the health of the Red Hat Ceph Storage cluster by running the **ceph health detail** command:

   Example

   ```
   [ceph: root@host01 /]# ceph health detail

   HEALTH_WARN 1 osds down; 1 OSDs or CRUSH {nodes, device-classes} have
   {NOUP,NODOWN,NOIN,NOOUT} flags set
   [WRN] OSD_DOWN: 1 osds down
      osd.1 (root=default,host=host01) is down
   [WRN] OSD_FLAGS: 1 OSDs or CRUSH {nodes, device-classes} have
   {NOUP,NODOWN,NOIN,NOOUT} flags set
      osd.1 has flags noup
   ```

   You can see that the storage cluster is in **HEALTH_WARN** status as one of the OSDs is down.

3. Mute the alert:

   Syntax

   ```
   ceph health mute HEALTH_MESSAGE
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph health mute OSD_DOWN
   ```

4. Optional: A health check mute can have a time to live (TTL) associated with it, such that the mute automatically expires after the specified period of time has elapsed. Specify the TTL as an optional duration argument in the command:

   Syntax

   ```
   ceph health mute HEALTH_MESSAGE DURATION
   ```

   DURATION can be specified in **s**, **sec**, **m**, **min**, **h**, or **hour**.

   Example

   ```
   [ceph: root@host01 /]# ceph health mute OSD_DOWN 10m
   ```

   In this example, the alert **OSD_DOWN** is muted for 10 minutes.

5. Verify if the Red Hat Ceph Storage cluster status has changed to **HEALTH_OK**:

   Example

   ```
   [ceph: root@host01 /]# ceph -s
     cluster:
       id:     81a4597a-b711-11eb-8cb8-001a4a000740
       health: HEALTH_OK
               (muted: OSD_DOWN(9m) OSD_FLAGS(9m))
   ```

```
  services:
    mon: 3 daemons, quorum host01,host02,host03 (age 33h)
    mgr: host01.pzhfuh(active, since 33h), standbys: host02.wsnngf, host03.xwzphg
    osd: 11 osds: 10 up (since 4m), 11 in (since 5d)

  data:
    pools:   1 pools, 1 pgs
    objects: 13 objects, 0 B
    usage:   85 MiB used, 165 GiB / 165 GiB avail
    pgs:     1 active+clean
```

In this example, you can see that the alert *OSD_DOWN* and *OSD_FLAG* is muted and the mute is active for nine minutes.

6. Optional: You can retain the mute even after the alert is cleared by making it **sticky**.

   Syntax

   ```
   ceph health mute HEALTH_MESSAGE DURATION --sticky
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph health mute OSD_DOWN 1h --sticky
   ```

7. You can remove the mute by running the following command:

   Syntax

   ```
   ceph health unmute HEALTH_MESSAGE
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph health unmute OSD_DOWN
   ```

Additional Resources

- See the *Health messages of a Ceph cluster* section in the *Red Hat Ceph Storage Troubleshooting Guide* for details.

1.5. UNDERSTANDING CEPH LOGS

Ceph stores its logs in the **/var/log/ceph/*CLUSTER_FSID*/** directory after the logging to files is enabled.

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.

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*.
- See the *Ceph daemon logs* to enable logging to files.

1.6. GENERATING AN SOS REPORT

You can run the **sos report** command to collect the configuration details, system information, and diagnostic information of a Red Hat Ceph Storage cluster from a Red Hat Enterprise Linux. Red Hat Support team uses this information for further troubleshooting of the storage cluster.

Prerequisites

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

Procedure

1. Install the **sos** package:

   #### Example

   ```
   [root@host01 ~]# dnf install sos
   ```

2. Run the **sos report** to get the system information of the storage cluster:

   #### Example

   ```
   [root@host01 ~]# sosreport -a --all-logs
   ```

   The report is saved in the /**var**/**tmp** file.

   Run the following command for specific Ceph daemon information:

   #### Example

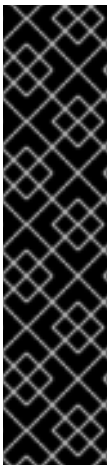
   ```
   [root@host01 ~]# sos report --all-logs -e
   ceph_mgr,ceph_common,ceph_mon,ceph_osd,ceph_ansible,ceph_mds,ceph_rgw
   ```

Additional Resources

- See the *What is an sosreport and how to create one in Red Hat Enterprise Linux?* KnowledgeBase article for more information.

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 2.1, "Ceph subsystems" for more information.

In addition, consider setting up a rotation of log files. See Section 2.4, "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 a 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 2.2, "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 2.3, "Configuring logging in configuration file" for details.

Prerequisites

- A running Red Hat Ceph Storage cluster.

2.1. 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/*CLUSTER_FSID*/** 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 a 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**.

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

```
2022-05-12 12:37:04.278761 7f45a9afc700 10 mon.cephn2@0(leader).osd e322 e322: 2 osds: 2 up,
2 in
2022-05-12 12:37:04.278792 7f45a9afc700 10 mon.cephn2@0(leader).osd e322
min_last_epoch_clean 322
2022-05-12 12:37:04.278795 7f45a9afc700 10 mon.cephn2@0(leader).log v1010106 log
2022-05-12 12:37:04.278799 7f45a9afc700 10 mon.cephn2@0(leader).auth v2877 auth
2022-05-12 12:37:04.278811 7f45a9afc700 20 mon.cephn2@0(leader) e1 sync_trim_providers
2022-05-12 12:37:09.278914 7f45a9afc700 11 mon.cephn2@0(leader) e1 tick
2022-05-12 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
2022-05-12 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
2022-05-12 12:37:09.278982 7f45a9afc700 10 mon.cephn2@0(leader).osd e322 e322: 2 osds: 2 up,
2 in
2022-05-12 12:37:09.278989 7f45a9afc700  5 mon.cephn2@0(leader).paxos(paxos active c
1028850..1029466) is_readable = 1 - now=2021-08-12 12:37:09.278990 lease_expire=0.000000 has
v0 lc 1029466
....
2022-05-12 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
2022-05-12 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
2022-05-12 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
....
2022-05-12 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

```
2022-05-12 11:27:53.869151 7f5d55d84700  1 -- 192.168.17.3:0/2410 --> 192.168.17.4:6801/2801 --
osd_ping(ping e322 stamp 2021-08-12 11:27:53.869147) v2 -- ?+0 0x63baa00 con 0x578dee0
2022-05-12 11:27:53.869214 7f5d55d84700  1 -- 192.168.17.3:0/2410 --> 192.168.0.114:6801/2801
-- osd_ping(ping e322 stamp 2021-08-12 11:27:53.869147) v2 -- ?+0 0x638f200 con 0x578e040
2022-05-12 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 2021-08-12
11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x63c1a00 con 0x578e040
2022-05-12 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 2021-08-12
11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x6313200 con 0x578dee0
....
```

```
2022-05-12 11:28:10.432313 7f5d6e71f700  5 osd.0 322 tick
2022-05-12 11:28:10.432375 7f5d6e71f700 20 osd.0 322 scrub_random_backoff lost coin flip,
randomly backing off
2022-05-12 11:28:10.432381 7f5d6e71f700 10 osd.0 322 do_waiters -- start
2022-05-12 11:28:10.432383 7f5d6e71f700 10 osd.0 322 do_waiters -- finish
```

Additional Resources

- *Configuring logging at runtime*
- *Configuring logging in configuration file*

2.2. 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:

   Syntax

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

Example

```
[ceph: root@host01 /]# 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 6.

2.3. 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 6

2.4. 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
```

2.5. CREATING AND COLLECTING OPERATION LOGS FOR CEPH OBJECT GATEWAY

User identity information is added to the operation log output. This is used to enable customers to access this information for auditing of S3 access. Track user identities reliably by S3 request in all versions of the Ceph Object Gateway operation log.

Procedure

1. Find where the logs are located:

   Syntax

   ```
   logrotate -f
   ```

   Example

   ```
   [root@host01 ~]# logrotate -f
   /etc/logrotate.d/ceph-12ab345c-1a2b-11ed-b736-fa163e4f6220
   ```

2. List the logs within the specified location:

   Syntax

   ```
   ll LOG_LOCATION
   ```

   Example

   ```
   [root@host01 ~]# ll /var/log/ceph/12ab345c-1a2b-11ed-b736-fa163e4f6220
    -rw-r--r--. 1 ceph ceph     412 Sep 28 09:26 opslog.log.1.gz
   ```

3. List the current buckets:

   Example

   ```
   [root@host01 ~]# /usr/local/bin/s3cmd ls
   ```

4. Create a bucket:

   Syntax

   ```
   /usr/local/bin/s3cmd mb s3://NEW_BUCKET_NAME
   ```

   Example

   ```
   [root@host01 ~]# /usr/local/bin/s3cmd mb s3://bucket1
   Bucket `s3://bucket1` created
   ```

5. List the current logs:

   Syntax

   ```
   ll LOG_LOCATION
   ```

   Example

   ```
   [root@host01 ~]# ll /var/log/ceph/12ab345c-1a2b-11ed-b736-fa163e4f6220
    total 852
    ...
   ```

```
-rw-r--r--. 1 ceph ceph    920 Jun 29 02:17 opslog.log
-rw-r--r--. 1 ceph ceph    412 Jun 28 09:26 opslog.log.1.gz
```

6. Collect the logs:

Syntax

```
tail -f LOG_LOCATION/opslog.log
```

Example

```
[root@host01 ~]# tail -f /var/log/ceph/12ab345c-1a2b-11ed-b736-fa163e4f6220/opslog.log

{"bucket":"","time":"2022-09-29T06:17:03.133488Z","time_local":"2022-09-
29T06:17:03.133488+0000","remote_addr":"10.0.211.66","user":"test1",
"operation":"list_buckets","uri":"GET /
HTTP/1.1","http_status":"200","error_code":"","bytes_sent":232,
"bytes_received":0,"object_size":0,"total_time":9,"user_agent":"","referrer":
"","trans_id":"tx00000c80881a9acd2952a-006335385f-175e5-primary",
"authentication_type":"Local","access_key_id":"1234","temp_url":false}

{"bucket":"cn1","time":"2022-09-29T06:17:10.521156Z","time_local":"2022-09-
29T06:17:10.521156+0000","remote_addr":"10.0.211.66","user":"test1",
"operation":"create_bucket","uri":"PUT /cn1/
HTTP/1.1","http_status":"200","error_code":"","bytes_sent":0,
"bytes_received":0,"object_size":0,"total_time":106,"user_agent":"",
"referrer":"","trans_id":"tx0000058d60c593632c017-0063353866-175e5-primary",
"authentication_type":"Local","access_key_id":"1234","temp_url":false}
```

CHAPTER 3. TROUBLESHOOTING NETWORKING ISSUES

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

Prerequisites

- A running Red Hat Ceph Storage cluster.

3.1. 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:

   Example

   ```
   [root@host01 ~]# dnf install net-tools
   [root@host01 ~]# dnf install telnet
   ```

2. Log into the **cephadm** shell and verify that the **public_network** parameters in the Ceph configuration file include the correct values:

   Example

   ```
   [ceph: root@host01 /]# cat /etc/ceph/ceph.conf
   # minimal ceph.conf for 57bddb48-ee04-11eb-9962-001a4a000672
   [global]
    fsid = 57bddb48-ee04-11eb-9962-001a4a000672
    mon_host = [v2:10.74.249.26:3300/0,v1:10.74.249.26:6789/0]
   [v2:10.74.249.163:3300/0,v1:10.74.249.163:6789/0]
   [v2:10.74.254.129:3300/0,v1:10.74.254.129:6789/0]
   [mon.host01]
   public network = 10.74.248.0/21
   ```

3. Exit the shell and verify that the network interfaces are up:

   Example

   ```
   [root@host01 ~]# 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: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode
DEFAULT group default qlen 1000
    link/ether 00:1a:4a:00:06:72 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@host01 ~]# ping host02
   ```

5. If you use a firewall, ensure that Ceph nodes are able to reach each 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@host01 ~]# firewall-cmd --info-zone=public
   public (active)
     target: default
     icmp-block-inversion: no
     interfaces: ens3
     sources:
     services: ceph ceph-mon cockpit dhcpv6-client ssh
     ports: 9283/tcp 8443/tcp 9093/tcp 9094/tcp 3000/tcp 9100/tcp 9095/tcp
     protocols:
     masquerade: no
     forward-ports:
     source-ports:
     icmp-blocks:
     rich rules:

   [root@host01 ~]# 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@host01 ~]# ethtool -S ens3 | 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@host01 ~]# ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.74.249.26  netmask 255.255.248.0  broadcast 10.74.255.255
        inet6 fe80::21a:4aff:fe00:672  prefixlen 64  scopeid 0x20<link>
        inet6 2620:52:0:4af8:21a:4aff:fe00:672  prefixlen 64  scopeid 0x0<global>
        ether 00:1a:4a:00:06:72  txqueuelen 1000  (Ethernet)
        RX packets 150549316  bytes 56759897541 (52.8 GiB)
        RX errors 0  dropped 176924  overruns 0  frame 0
        TX packets 55584046  bytes 62111365424 (57.8 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 9373290  bytes 16044697815 (14.9 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9373290  bytes 16044697815 (14.9 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

c. Using the **netstat** command:

Example

```
[root@host01 ~]# netstat -ai
Kernel Interface table
Iface             MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-
OVR Flg
ens3             1500 311847720      0 364903 0      114341918      0      0      0 BMRU
lo              65536 19577001       0      0 0       19577001       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:

Example

```
[root@host01 ~]# dnf install iperf3
```

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

Example

```
[root@host01 ~]# 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@host02 ~]# 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@host01 ~]# ethtool ens3
Settings for ens3:
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 *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.
- For more information, see *Knowledgebase articles and solutions* related to troubleshooting networking issues on the Customer Portal.

3.2. BASIC CHRONY NTP TROUBLESHOOTING

This section includes basic chrony NTP 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.

CHAPTER 4. TROUBLESHOOTING CEPH MONITORS

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

Prerequisites

- Verify the network connection.

4.1. 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.

Prerequisites

- A running Red Hat Ceph Storage cluster.

4.1.1. 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

4.1.2. 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

4.1.3. 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:

   Syntax

```
systemctl status ceph-FSID@DAEMON_NAME
systemctl start ceph-FSID@DAEMON_NAME
```

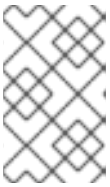
Example

```
[root@mon ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
[root@mon ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
```

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 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 located at **/var/log/ceph/*CLUSTER_FSID*/ceph-mon.*HOST_NAME*.log** by default.



NOTE

By default, the monitor logs are not present in the log folder. You need to enable logging to files for the logs to appear in the folder. See the *Ceph daemon logs* to enable logging to files.

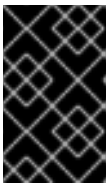
2. If the log contains error messages similar to the following ones, the Ceph Monitor might have a corrupted store.

```
Corruption: error in middle of record
Corruption: 1 missing files; example: /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:

```
[ceph: root@host01 /]# ceph daemon mon.host01 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.

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* section in the *Red Hat Ceph Storage Administration Guide*.
- The *Using the Ceph Administration Socket* section in the *Red Hat Ceph Storage Administration Guide*.

4.1.4. 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)
```

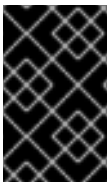
In addition, Ceph logs contain error messages similar to these:

```
2022-05-04 07:28:32.035795 7f806062e700 0 log [WRN] : mon.a 127.0.0.1:6789/0 clock skew 0.14s
> max 0.05s
2022-05-04 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.



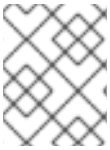
IMPORTANT

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 chrony 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.
2. If you use a remote NTP server, consider deploying your own chrony NTP server on your network. For details, see the *Using the Chrony Suite to Configure NTP* chapter within the *Configuring basic system settings* guide within the Product Documentation for {os-product for your OS version, on the Red Hat Customer Portal.



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*

4.1.5. 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 RocksDB database that stores entries as key–values pairs. The database includes a cluster map and is located by default at **/var/lib/ceph/*CLUSTER_FSID*/mon.*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:

Syntax

```
du -sch /var/lib/ceph/CLUSTER_FSID/mon.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

```
[root@mon ~]# du -sh  /var/lib/ceph/b341e254-b165-11ed-a564-ac1f6bb26e8c/mon.host01/
109M /var/lib/ceph/b341e254-b165-11ed-a564-ac1f6bb26e8c/mon.host01/
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

4.1.6. 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 a longer time.

Additional Resources

- For details, see the *Using the Ceph Administration Socket* section in the *Administration Guide* for Red Hat Ceph Storage 6.
- See the Section 4.1.1, "Ceph Monitor error messages" in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the Section 4.1.2, "Common Ceph Monitor error messages in the Ceph logs" in the *Red Hat Ceph Storage Troubleshooting Guide*.

4.2. 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:

   Example

   ```
   [ceph: root@host01 /]# 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:

      Syntax

      ```
      systemctl stop ceph-FSID@DAEMON_NAME
      ```

      Example

      ```
      [root@mon ~]# systemctl stop ceph-b404c440-9e4c-11ec-a28a-
      001a4a0001df@mon.host01.service
      ```

   b. Copy the Ceph Monitor map:

      Syntax

      ```
      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:

Example

```
[ceph: root@host01 /]#  ceph-mon -i mon.a  --extract-monmap /tmp/monmap
```

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

Syntax

```
systemctl stop ceph-FSID@DAEMON_NAME
```

Example

```
[root@mon ~]# systemctl stop ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
```

4. Inject the Ceph Monitor map:

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.host01 --inject-monmap /tmp/monmap
```

5. Start the Ceph Monitor:

Syntax

```
systemctl start ceph-FSID@DAEMON_NAME
```

Example

```
[root@mon ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
```

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

Syntax

```
systemctl start ceph-FSID@DAEMON_NAME
```

Example

```
[root@mon ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
```

Additional Resources

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

4.3. REPLACING A FAILED MONITOR

When a Ceph Monitor has a corrupted store, you can replace the monitor in the storage cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Able to form a quorum.
- 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**:

   ```
   [ceph: root@host01 /]# ceph mon remove host01 --cluster remote
   ```

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

Additional Resources

- See the *Ceph Monitor is out of quorum* for details.

4.4. 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:

   Syntax

   ```
   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.

   Example

   ```
   [ceph: root@host01 /]# ceph tell mon.host01 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:

   Syntax

   ```
   systemctl restart ceph-FSID@DAEMON_NAME
   ```

   Example

   ```
   [root@mon ~]# systemctl restart ceph-b404c440-9e4c-11ec-a28a-
   001a4a0001df@mon.host01.service
   ```

5. Ensure that Monitors have formed a quorum:

   ```
   [ceph: root@host01 /]# 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.

   Syntax

   ```
   systemctl status ceph-FSID@DAEMON_NAME
   systemctl stop ceph-FSID@DAEMON_NAME
   ```

   Example

   ```
   [root@mon ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-001a4a0001df@mon.host01.service
   [root@mon ~]# systemctl stop ceph-b404c440-9e4c-11ec-a28a-001a4a0001df@mon.host01.service
   ```

8. Compact the Monitor store:

   Syntax

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

   Replace ***HOST_NAME*** with a short host name of the Monitor host.

   Example

   ```
   [ceph: root@host01 /]# ceph-monstore-tool /var/lib/ceph/b404c440-9e4c-11ec-a28a-001a4a0001df/mon.host01 compact
   ```

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

   Syntax

   ```
   systemctl start ceph-FSID@DAEMON_NAME
   ```

   Example

   ```
   [root@mon ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-001a4a0001df@mon.host01.service
   ```

Additional Resources

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

4.5. 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.

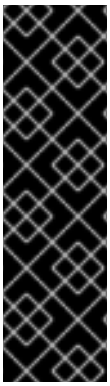
4.6. RECOVERING THE CEPH MONITOR STORE

Ceph Monitors store the cluster map in a key-value store such as RocksDB. 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
```

The 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.

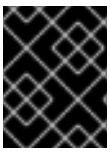
If there is corruption on all Ceph Monitors, you can recover it with information stored on the OSD nodes by using utilities called **ceph-monstore-tool** and **ceph-objectstore-tool**.



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.

4.6.1. 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 assistance with the recovering process.

Prerequisites

- 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. Mount all disks with Ceph data to a temporary location. Repeat this step for all OSD nodes.
   a. List the data partitions using the **ceph-volume** command:

      Example

      ```
      [ceph: root@host01 /]# ceph-volume lvm list
      ```

   b. Mount the data partitions to a temporary location:

      Syntax

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

   c. Restore the SELinux context:

      Syntax

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

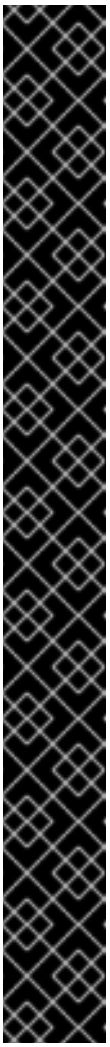
      Replace *OSD_ID* with a numeric, space-separated list of Ceph OSD IDs on the OSD node.

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

Syntax

```
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:

   Syntax

   ```
   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 OSD data and *OSD-ID* with the ID of the OSD.

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

   Syntax

   ```
   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:

   Example

   ```
   [root@host01 ~]# cd /root/
   [root@host01 ~]# ms=/tmp/monstore/
   [root@host01 ~]# db=/root/db/
   [root@host01 ~]# db_slow=/root/db.slow/

   [root@host01 ~]# mkdir $ms
   [root@host01 ~]# 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:

Example

```
[ceph: root@host01 /]# ceph-authtool /etc/ceph/ceph.client.admin.keyring -n mon. --cap
mon 'allow *' --gen-key
[ceph: root@host01 /]# 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:

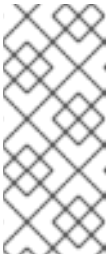
Example

```
[ceph: root@host01 /]# mv /root/db/*.sst /root/db.slow/*.sst /tmp/monstore/store.db
```

d. Rebuild the Monitor store from the collected map:

Example

```
[ceph: root@host01 /]# 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:

Syntax

```
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:

Syntax

```
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:

Syntax

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

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

3. Unmount all the temporary mounted OSDs on all nodes:

Example

```
[root@host01 ~]# umount /var/lib/ceph/osd/ceph-*
```

4. Start all the Ceph Monitor daemons:

Syntax

```
systemctl start ceph-FSID@DAEMON_NAME
```

Example

```
[root@mon ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@mon.host01.service
```

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

Syntax

```
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:

Syntax

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

Example

```
[root@mon ~]# systemctl start ceph-b341e254-b165-11ed-a564-
ac1f6bb26e8c@mgr.extensa003.exrqql.service
```

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

7. Start all OSD processes across all OSD nodes. Repeat for all OSDs on the cluster:

Syntax

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

Example

```
[root@host01 ~]# systemctl start ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@osd.0.service
```

8. Ensure that the OSDs are returning to service:

Example

```
[ceph: root@host01 /]# ceph -s
```

Additional Resources

- For details on registering Ceph nodes to the Content Delivery Network (CDN), see *Registering the Red Hat Ceph Storage nodes to the CDN and attaching subscriptions* section in the *Red Hat Ceph Storage Installation Guide*.
- See *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.

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.

5.1. 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.

Prerequisites

- Root-level access to the Ceph OSD nodes.

5.1.1. 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	Down OSDs Flapping OSDs
requests are blocked	Slow request or requests are blocked
slow requests	Slow request or requests are blocked

5.1.2. 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

5.1.3. 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.

5.1.4. 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.

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

      ```
      [ceph: root@host01 /]# ceph osd df
      ```

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

      ```
      df
      ```

   c. If needed, add a new OSD node.

Additional Resources

- *Full OSDs*
- See the *Set an OSD's Weight by Utilization* section in the *Storage Strategies* guide for Red Hat Ceph Storage 6.
- For details, see the *CRUSH Tunables* section in the *Storage Strategies* guide for Red Hat Ceph Storage 6 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.

5.1.6. Down OSDs

The **ceph health detail** 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**:

```
[ceph: root@host01 /]# 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. Replace the *OSD_ID* with the ID of the OSD that is down:

   Syntax

```
systemctl restart ceph-FSID@osd.OSD_ID
```

   Example

```
[root@host01 ~]# systemctl restart ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@osd.0.service
```

   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.

   a. Check the corresponding log file to determine the cause of the failure. By default, Ceph stores log files in the **/var/log/ceph/*CLUSTER_FSID*/** directory after the logging to files is enabled.

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

   c. 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")
   ```

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

   ```
   dmesg
   ```

   a. The **error -5** error message similar to the following one indicates corruption of the underlying XFS file system. For details on how to fix this problem, see the *What is the meaning of "xfs_log_force: error -5 returned"?* solution on the Red Hat Customer Portal.

   ```
   xfs_log_force: error -5 returned
   ```

   b. 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.

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

6. 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/*CLUSTER_FSID*/** directory after the logging to files is enabled.

   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 *Ceph daemon logs* to enable logging to files.

5.1.7. 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
2022-05-05 06:27:20.810535 mon.0 [INF] osdmap e609: 9 osds: 8 up, 9 in
2022-05-05 06:27:24.120611 mon.0 [INF] osdmap e611: 9 osds: 7 up, 9 in
2022-05-05 06:27:25.975622 mon.0 [INF] HEALTH_WARN; 118 pgs stale; 2/9 in osds are down
2022-05-05 06:27:27.489790 mon.0 [INF] osdmap e614: 9 osds: 6 up, 9 in
2022-05-05 06:27:36.540000 mon.0 [INF] osdmap e616: 9 osds: 7 up, 9 in
2022-05-05 06:27:39.681913 mon.0 [INF] osdmap e618: 9 osds: 8 up, 9 in
2022-05-05 06:27:43.269401 mon.0 [INF] osdmap e620: 9 osds: 9 up, 9 in
2022-05-05 06:27:54.884426 mon.0 [INF] osdmap e622: 9 osds: 8 up, 9 in
2022-05-05 06:27:57.398706 mon.0 [INF] osdmap e624: 9 osds: 7 up, 9 in
2022-05-05 06:27:59.669841 mon.0 [INF] osdmap e625: 9 osds: 6 up, 9 in
2022-05-05 06:28:07.043677 mon.0 [INF] osdmap e628: 9 osds: 7 up, 9 in
2022-05-05 06:28:10.512331 mon.0 [INF] osdmap e630: 9 osds: 8 up, 9 in
2022-05-05 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:

```
2022-05-25 03:44:06.510583 osd.50 127.0.0.1:6801/149046 18992 : cluster [WRN] map e600547
wrongly marked me down
```

```
2022-05-25 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 the network.

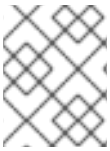
Ceph OSDs cannot manage situations where the private network for the storage cluster fails, or significant latency is on the public client-facing network.

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 the 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.

4. Alternatively, you can temporarily 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 *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*.

5.1.8. 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:

```
2022-05-24 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
```

```
2022-05-25 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:

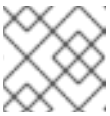
- Problems with the underlying hardware, such as disk drives, hosts, racks, or network switches
- Problems with the network. These problems are usually connected with flapping OSDs. See *Flapping OSDs* for details.
- System load

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 6.

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

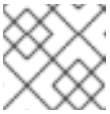
1. Determine if the OSDs with slow or block requests share a common piece of hardware, for example, a disk drive, host, rack, or network switch.
2. If the OSDs share a disk:
   a. Use the **smartmontools** utility to check the health of the disk or the logs to determine any errors on the disk.



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.
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.

5.2. 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. Log in to the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Set the **noout** flag before stopping the OSD:

   Example

   ```
   [ceph: root@host01 /]# ceph osd set noout
   ```

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

   Example

   ```
   [ceph: root@host01 /]# ceph osd unset noout
   ```

Additional Resources

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

5.3. 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

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

Removing an OSD from the Ceph Cluster

1. Log into the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Determine which OSD is **down**.

   Example

   ```
   [ceph: root@host01 /]# ceph osd tree | grep -i down
   ID  CLASS  WEIGHT   TYPE NAME          STATUS  REWEIGHT  PRI-AFF
    0   hdd 0.00999        osd.0      down  1.00000          1.00000
   ```

3. Mark the OSD as **out** for the cluster to rebalance and copy its data to other OSDs.

   Syntax

   ```
   ceph osd out OSD_ID.
   ```

   Example

   ```
   [ceph: root@host01 /]# 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 based on the **mon_osd_down_out_interval** parameter. 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.

4. Ensure that the failed OSD is backfilling.

   Example

   ```
   [ceph: root@host01 /]# ceph -w | grep backfill
   2022-05-02 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
   2022-05-02 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
   2022-05-02 04:48:00.380063 osd.1 [INF] 0.6f starting backfill to osd.0 from (0'0,0'0] MAX to
   2521'166639
   2022-05-02 04:48:00.380139 osd.1 [INF] 0.48 starting backfill to osd.0 from (0'0,0'0] MAX to
   2513'43079
   2022-05-02 04:48:00.380260 osd.1 [INF] 0.d starting backfill to osd.0 from (0'0,0'0] MAX to
   2513'136847
   2022-05-02 04:48:00.380849 osd.1 [INF] 0.71 starting backfill to osd.0 from (0'0,0'0] MAX to
   2331'28496
   2022-05-02 04:48:00.381027 osd.1 [INF] 0.51 starting backfill to osd.0 from (0'0,0'0] MAX to
   2513'87544
   ```

   You should see the placement group states change from **active+clean** to **active**, some degraded objects, and finally **active+clean** when migration completes.

5. Stop the OSD:

   Syntax

   ```
   ceph orch daemon stop OSD_ID
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph orch daemon stop osd.0
   ```

6. Remove the OSD from the storage cluster:

   Syntax

```
ceph orch osd rm OSD_ID --replace
```

Example

```
[ceph: root@host01 /]# ceph orch osd rm 0 --replace
```

The *OSD_ID* is preserved.

Replacing the physical drive

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

1. If the drive is hot-swappable, replace the failed drive with a new one.
2. 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.
3. When the drive appears under the /**dev**/ directory, make a note of the drive path.
4. If you want to add the OSD manually, find the OSD drive and format the disk.

Adding an OSD to the Ceph Cluster

1. Once the new drive is inserted, you can use the following options to deploy the OSDs:
   - The OSDs are deployed automatically by the Ceph Orchestrator if the **--unmanaged** parameter is not set.

     Example

     ```
     [ceph: root@host01 /]# ceph orch apply osd --all-available-devices
     ```

   - Deploy the OSDs on all the available devices with the **unmanaged** parameter set to **true**.

     Example

     ```
     [ceph: root@host01 /]# ceph orch apply osd --all-available-devices --unmanaged=true
     ```

   - Deploy the OSDs on specific devices and hosts.

     Example

     ```
     [ceph: root@host01 /]# ceph orch daemon add osd host02:/dev/sdb
     ```

2. Ensure that the CRUSH hierarchy is accurate:

   Example

   ```
   [ceph: root@host01 /]# ceph osd tree
   ```

Additional Resources

- See the *Deploying Ceph OSDs on all available devices* section in the *Red Hat Ceph Storage Operations Guide*.
- See the *Deploying Ceph OSDs on specific devices and hosts* section in the *Red Hat Ceph Storage Operations Guide*.
- See the *Down OSDs* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Red Hat Ceph Storage Installation Guide*.

5.4. 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
```

5.5. 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** afterward. To temporarily increase the **full_ratio** value, increase the **set-full-ratio** instead.

Prerequisites

- Root-level access to the Ceph Monitor node.

Procedure

1. Log in to the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

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

```
[ceph: root@host01 /]# ceph osd dump | grep -i full
full_ratio 0.95
```

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

```
[ceph: root@host01 /]# 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.

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

```
[ceph: root@host01 /]# ceph osd dump | grep -i full
full_ratio 0.97
```

5. Monitor the cluster state:

```
[ceph: root@host01 /]# ceph -w
```

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

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

```
[ceph: root@host01 /]# ceph osd set-full-ratio 0.95
```

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

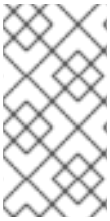
```
[ceph: root@host01 /]# ceph osd dump | grep -i full
full_ratio 0.95
```

Additional Resources

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

CHAPTER 6. TROUBLESHOOTING A MULTI-SITE CEPH OBJECT GATEWAY

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



NOTE

When the **radosgw-admin bucket sync status** command reports that the bucket is behind on shards even if the data is consistent across multi-site, run additional writes to the bucket. It synchronizes the status reports and displays a message that the bucket is caught up with source.

Prerequisites

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

6.1. 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.

ERROR: request failed: (13) Permission denied If the realm has been changed on the master zone, the master zone's gateway may need to be restarted to recognize this user

While configuring the secondary site, sometimes a **rgw realm pull --url http://primary_endpoint --access-key <> --secret <>** fails with a permission denied error.
In such cases, ensure that on the primary site, the system user credentials are the same via the following commands:

```
radosgw-admin user info --uid synchronization_user, and
radosgw-admin zone get
```

Additional Resources

- Contact Red Hat Support for any additional assistance.

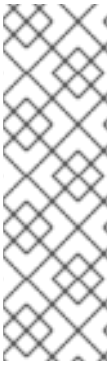
6.2. SYNCING A MULTI-SITE CEPH OBJECT GATEWAY

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

Example

```
[ceph: root@host01 /]# 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

```
[ceph: root@host01 /]# radosgw-admin data sync status --shard-id=27 --source-zone=us-east
```

```
{
  "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*.

Syntax

```
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.

6.3. 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 --admin-daemon** command to view the current metric data for the performance counters:

Syntax

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

Example

```
[ceph: root@host01 /]# ceph --admin-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 --admin-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.

6.4. SYNCHRONIZING DATA IN A MULTI-SITE CEPH OBJECT GATEWAY CONFIGURATION

In a multi-site Ceph Object Gateway configuration of a storage cluster, failover and failback causes data synchronization to stop. The **radosgw-admin sync status** command reports that the data sync is behind for an extended period of time.

You can run the **radosgw-admin data sync init** command to synchronize data between the sites and then restart the Ceph Object Gateway. This command does not touch any actual object data and initiates data sync for a specified source zone. It causes the zone to restart a full sync from the source zone.

IMPORTANT

Contact *Red Hat support* before running the **data sync init** command.

If you are going for a full restart of sync, and if there is a lot of data that needs to be synced on the source zone, then the bandwidth consumption is high and then you have to plan accordingly.

NOTE

If a user accidentally deletes a bucket on the secondary site, you can use the **metadata sync init** command on the site to synchronize data.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Ceph Object Gateway configured at two sites at least.

Procedure

1. Check the sync status between the sites:

   Example

   ```
   [ceph: host04 /]# radosgw-admin sync status
             realm d713eec8-6ec4-4f71-9eaf-379be18e551b (india)
         zonegroup ccf9e0b2-df95-4e0a-8933-3b17b64c52b7 (shared)
              zone 04daab24-5bbd-4c17-9cf5-b1981fd7ff79 (primary)
     current time 2022-09-15T06:53:52Z
   zonegroup features enabled: resharding
     metadata sync no sync (zone is master)
         data sync source: 596319d2-4ffe-4977-ace1-8dd1790db9fb (secondary)
                           syncing
                           full sync: 0/128 shards
                           incremental sync: 128/128 shards
                           data is caught up with source
   ```

2. Synchronize data from the secondary zone:

   Example

   ```
   [ceph: root@host04 /]# radosgw-admin data sync init --source-zone primary
   ```

3. Restart all the Ceph Object Gateway daemons at the site:

   Example

   ```
   [ceph: root@host04 /]# ceph orch restart rgw.myrgw
   ```

CHAPTER 7. TROUBLESHOOTING CEPH PLACEMENT GROUPS

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

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.

7.1. MOST COMMON CEPH PLACEMENT GROUPS ERRORS

The following table lists the most common error 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 7.2, "Listing placement groups stuck in **stale**, **inactive**, or **unclean** state" for details.

Prerequisites

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

7.1.1. 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

7.1.2. 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. By 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 includes information similar to the following example:

   Example

   ```
   [ceph: root@host01 /]# 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 6

7.1.3. Inconsistent placement groups

Some placement groups are marked as **active + clean + inconsistent** and the **ceph health detail** returns an error message 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. Log in to the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Determine which placement group is in the **inconsistent** state:

   ```
   [ceph: root@host01 /]# 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
   ```

3. Determine why the placement group is **inconsistent**.

   a. Start the deep scrubbing process on the placement group:

      Syntax

      ```
      ceph pg deep-scrub ID
      ```

      Replace ***ID*** with the ID of the **inconsistent** placement group, for example:

      ```
      [ceph: root@host01 /]# 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:

      Syntax

      ```
      ceph -w | grep ID
      ```

      Replace ***ID*** with the ID of the **inconsistent** placement group, for example:

      ```
      [ceph: root@host01 /]# ceph -w | grep 0.6
      2022-05-26 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.
      2022-05-26 01:35:36.788334 osd.106 [ERR] 0.6 deep-scrub 1 errors
      ```

4. 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.

Syntax

```
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
```

5. 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

- See the *Listing placement group inconsistencies* in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Ceph data integrity* section in the *Red Hat Ceph Storage Architecture Guide*.
- See the *Scrubbing the OSD* section in the *Red Hat Ceph Storage Configuration Guide*.

7.1.4. 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: root@host01 /]# 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* .

7.1.5. Inactive placement groups

The **ceph health** command returns an 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.

7.1.6. 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:

Syntax

```
ceph pg ID query
```

Replace ***ID*** with the ID of the placement group that is **down**:

Example

```
[ceph: root@host01 /]#  ceph pg 0.5 query

{ "state": "down+peering",
  ...
  "recovery_state": [
      { "name": "Started\/Primary\/Peering\/GetInfo",
        "enter_time": "2021-08-06 14:40:16.169679",
        "requested_info_from": []},
      { "name": "Started\/Primary\/Peering",
        "enter_time": "2021-08-06 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": "2021-08-06 14:40:16.169513"}
  ]
}
```

The **recovery_state** section includes information on 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*.

7.1.7. 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. Log in to the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Determine which placement group contains **unfound** objects:

   ```
   [ceph: root@host01 /]# 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%)**
   ```

3. List more information about the placement group:

   Syntax

   ```
   ceph pg ID query
   ```

   Replace ***ID*** with the ID of the placement group containing the **unfound** objects:

   Example

   ```
   [ceph: root@host01 /]# 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": "2021-08-28 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": "2021-08-28 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.

4. Troubleshoot the OSDs that are marked as **down**. See *Down OSDs* for details.
5. 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.

7.2. 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 report 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
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. Log into the Cephadm shell:

   Example

   ```
   [root@host01 ~]# cephadm shell
   ```

2. List the stuck PGs:

   Example

   ```
   [ceph: root@host01 /]# ceph pg dump_stuck inactive
   [ceph: root@host01 /]# ceph pg dump_stuck unclean
   [ceph: root@host01 /]# ceph pg dump_stuck stale
   ```

Additional Resources

- See the *Placement Group States* section in the *Red Hat Ceph Storage Administration Guide*.

7.3. LISTING PLACEMENT GROUP INCONSISTENCIES

Use the **rados** utility to list inconsistencies in various replicas of 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

- List all the inconsistent placement groups in a pool:

  #### Syntax

  ```
  rados list-inconsistent-pg POOL --format=json-pretty
  ```

  #### Example

  ```
  [ceph: root@host01 /]# rados list-inconsistent-pg data --format=json-pretty
  [0.6]
  ```

- List inconsistent objects in a placement group with ID:

  #### Syntax

  ```
  rados list-inconsistent-obj PLACEMENT_GROUP_ID
  ```

  #### Example

  ```
  [ceph: root@host01 /]# 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.

- List inconsistent sets of snapshots:

  Syntax

  ```
  rados list-inconsistent-snapset PLACEMENT_GROUP_ID
  ```

  Example

  ```
  [ceph: root@host01 /]# 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*.

7.4. 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

- Repair the **inconsistent** placement groups:

  Syntax

  ```
  ceph pg repair ID
  ```

  Replace ***ID*** with the ID of the **inconsistent** placement group.

Additional Resources

- See the *Inconsistent placement groups* section in the *Red Hat Ceph Storage Troubleshooting Guide*.
- See the *Listing placement group inconsistencies* *Red Hat Ceph Storage Troubleshooting Guide*.

7.5. 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 a 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:

```
[ceph: root@host01 /]# 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:

```
[ceph: root@host01 /]# ceph osd set noscrub
[ceph: root@host01 /]# 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:

   Syntax

   ```
   ceph osd pool set POOL pg_num VALUE
   ```

   Specify the pool name and the new value, for example:

   Example

   ```
   [ceph: root@host01 /]# ceph osd pool set data pg_num 4
   ```

   c. Monitor the status of the cluster:

   Example

   ```
   [ceph: root@host01 /]# 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:

   Syntax

   ```
   ceph osd pool set POOL pgp_num VALUE
   ```

   Specify the pool name and the new value, for example:

   ```
   [ceph: root@host01 /]# ceph osd pool set data pgp_num 4
   ```

c. Monitor the status of the cluster:

```
[ceph: root@host01 /]# 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: root@host01 /]# 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: root@host01 /]# ceph osd unset noscrub
[ceph: root@host01 /]# ceph osd unset nodeep-scrub
```

Additional Resources

- See the *Nearfull OSDs*
- See the *Monitoring Placement Group Sets* section in the *Red Hat Ceph Storage Administration Guide*.
- 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.
- See the *Auto-scaling placement groups* section in the Red Hat Ceph Storage Storage Strategies Guide for more information on PG autoscaler.

CHAPTER 8. 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.

IMPORTANT

Manipulating objects can cause unrecoverable data loss. Contact Red Hat support before using the **ceph-objectstore-tool** utility.

Prerequisites

- Verify there are no network-related issues.

8.1. 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.

Prerequisites

- Root-level access to the Ceph OSD nodes.

8.1.1. 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:

   Syntax

   ```
   systemctl status ceph-FSID@osd.OSD_ID
   ```

Example

```
[root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@osd.0.service
```

2. Log in to the OSD container:

Syntax

```
cephadm shell --name osd.OSD_ID
```

Example

```
[root@host01 ~]# cephadm shell --name osd.0
```

3. Identify all the objects within an OSD, regardless of their placement group:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --op list
```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list
```

4. Identify all the objects within a placement group:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID --op list
```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c
--op list
```

5. Identify the PG an object belongs to:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --op list OBJECT_ID
```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op list
default.region
```

8.1.2. 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

   ```
   systemctl status ceph-FSID@osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-001a4a0001df@osd.0.service
   ```

2. Log in to the OSD container:

   Syntax

   ```
   cephadm shell --name osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# cephadm shell --name osd.0
   ```

3. To list all the lost legacy objects:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost --dry-run
   ```

   Example

   ```
   [ceph: root@host01 /]# 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. Select the appropriate circumstance:

   a. To fix all lost objects:

      Syntax

      ```
      ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost
      ```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op fix-lost
```

b. To fix all the lost objects within a placement group:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID --op fix-lost
```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --pgid 0.1c --op fix-lost
```

c. To fix a lost object by its identifier:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --op fix-lost OBJECT_ID
```

Example

```
[ceph: root@host01 /]# ceph-objectstore-tool --data-path /var/lib/ceph/osd/ceph-0 --op fix-lost default.region
```

8.2. 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.

Prerequisites

- Root-level access to the Ceph OSD nodes.

8.2.1. 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:

   Syntax

   ```
   systemctl status ceph-FSID@osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
   001a4a0001df@osd.0.service
   ```

2. Log in to the OSD container:

   Syntax

   ```
   cephadm shell --name osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# cephadm shell --name osd.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:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
   OBJECT \
   get-bytes > OBJECT_FILE_NAME
   ```

   Example

```
[ceph: root@host01 /]# 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

[ceph: root@host01 /]#  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:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
set-bytes < OBJECT_FILE_NAME
```

Example

```
[ceph: root@host01 /]# 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
```

8.2.2. 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. Log in to the OSD container:

Syntax

```
cephadm shell --name osd.OSD_ID
```

Example

```
[root@host01 ~]# cephadm shell --name osd.0
```

2. Remove an object:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
remove
```

Example

```
[ceph: root@host01 /]# 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
```

8.2.3. 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:

Syntax

```
systemctl status ceph-osd@OSD_ID
```

Example

```
[root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@osd.0.service
```

2. Log in to the OSD container:

Syntax

```
cephadm shell --name osd.OSD_ID
```

Example

```
[root@host01 ~]# cephadm shell --name osd.0
```

3. List the object map:

Syntax

```
ceph-objectstore-tool --data-path PATH_TO_OSD --pgid PG_ID \
OBJECT \
list-omap
```

Example

```
[ceph: root@host01 /]# 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
```

8.2.4. Manipulating the object map header

The **ceph-objectstore-tool** utility outputs 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. Verify the appropriate OSD is down:

Syntax

```
systemctl status ceph-FSID@osd.OSD_ID
```

Example

```
[root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
001a4a0001df@osd.0.service
```

2. Log in to the OSD container:

Syntax

```
cephadm shell --name osd.OSD_ID
```

Example

```
[root@host01 ~]# cephadm shell --name osd.0
```

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

```
[ceph: root@host01 /]# 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

```
[ceph: root@host01 /]# 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
```

8.2.5. 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. Log in to the OSD container:

Syntax

```
cephadm shell --name osd.OSD_ID
```

Example

```
[root@host01 ~]# cephadm shell --name osd.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

```
[ceph: root@host01 /]# 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
```

- 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

```
[ceph: root@host01 /]# 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

```
[ceph: root@host01 /]# 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 ""
```

8.2.6. 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:

   Syntax

   ```
   systemctl status ceph-FSID@osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
   001a4a0001df@osd.0.service
   ```

2. Log in to the OSD container:

   Syntax

   ```
   cephadm shell --name osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# cephadm shell --name osd.0
   ```

3. List the object's attributes:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD \
   --pgid PG_ID OBJECT \
   list-attrs
   ```

   Example

   ```
   [ceph: root@host01 /]# 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
   ```

8.2.7. 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 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.

- Stop the **ceph-osd** daemon.

Procedure

1. Verify the appropriate OSD is down:

   Syntax

   ```
   systemctl status ceph-FSID@osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# systemctl status ceph-b404c440-9e4c-11ec-a28a-
   001a4a0001df@osd.0.service
   ```

2. Log in to the OSD container:

   Syntax

   ```
   cephadm shell --name osd.OSD_ID
   ```

   Example

   ```
   [root@host01 ~]# cephadm shell --name osd.0
   ```

3. Get the object's attributes:

   Syntax

   ```
   ceph-objectstore-tool --data-path PATH_TO_OSD \
   --pgid PG_ID OBJECT \
   get-attr KEY > OBJECT_ATTRS_FILE_NAME
   ```

   Example

   ```
   [ceph: root@host01 /]# 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-attr "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-attr KEY < OBJECT_ATTRS_FILE_NAME
   ```

   Example

   ```
   [ceph: root@host01 /]# 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-attr "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-attr KEY
```

Example

```
[ceph: root@host01 /]# 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-attr "oid"
```

Additional Resources

- For Red Hat Ceph Storage support, see the Red Hat Customer Portal.

CHAPTER 9. TROUBLESHOOTING CLUSTERS IN STRETCH MODE

You can replace and remove the failed tiebreaker monitors. You can also force the cluster into the recovery or healthy mode if needed.

Additional Resources

See *Stretch clusters for Ceph storage* for more information about clusters in stretch mode.

9.1. REPLACING THE TIEBREAKER WITH A MONITOR IN QUORUM

If your tiebreaker monitor fails, you can replace it with an existing monitor in quorum and remove it from the cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster
- Stretch mode is enabled on a cluster

Procedure

1. Disable automated monitor deployment:

   Example

   ```
   [ceph: root@host01 /]# ceph orch apply mon --unmanaged

   Scheduled mon update…
   ```

2. View the monitors in quorum:

   Example

   ```
   [ceph: root@host01 /]# ceph -s

   mon: 5 daemons, quorum host01, host02, host04, host05 (age 30s), out of quorum: host07
   ```

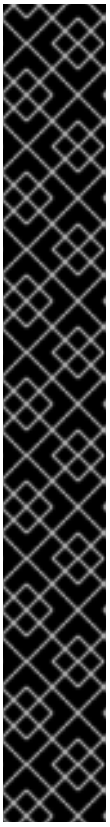
3. Set the monitor in quorum as a new tiebreaker:

   Syntax

   ```
   ceph mon set_new_tiebreaker NEW_HOST
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph mon set_new_tiebreaker host02
   ```



IMPORTANT

You get an error message if the monitor is in the same location as existing non-tiebreaker monitors:

Example

```
[ceph: root@host01 /]# ceph mon set_new_tiebreaker host02

Error EINVAL: mon.host02 has location DC1, which matches mons host02 on the datacenter dividing bucket for stretch mode.
```

If that happens, change the location of the monitor:

Syntax

```
ceph mon set_location HOST datacenter=DATACENTER
```

Example

```
[ceph: root@host01 /]# ceph mon set_location host02 datacenter=DC3
```

4. Remove the failed tiebreaker monitor:

   Syntax

   ```
   ceph orch daemon rm FAILED_TIEBREAKER_MONITOR --force
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph orch daemon rm mon.host07 --force

   Removed mon.host07 from host 'host07'
   ```

5. Once the monitor is removed from the host, redeploy the monitor:

   Syntax

   ```
   ceph mon add HOST IP_ADDRESS datacenter=DATACENTER
   ceph orch daemon add mon HOST
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph mon add host07 213.222.226.50 datacenter=DC1
   [ceph: root@host01 /]# ceph orch daemon add mon host07
   ```

6. Ensure there are five monitors in quorum:

   Example

```
[ceph: root@host01 /]# ceph -s

mon: 5 daemons, quorum host01, host02, host04, host05, host07 (age 15s)
```

7. Verify that everything is configured properly:

   Example

```
[ceph: root@host01 /]# ceph mon dump

epoch 19
fsid 1234ab78-1234-11ed-b1b1-de456ef0a89d
last_changed 2023-01-17T04:12:05.709475+0000
created 2023-01-16T05:47:25.631684+0000
min_mon_release 16 (pacific)
election_strategy: 3
stretch_mode_enabled 1
tiebreaker_mon host02
disallowed_leaders host02
0: [v2:132.224.169.63:3300/0,v1:132.224.169.63:6789/0] mon.host02; crush_location
{datacenter=DC3}
1: [v2:220.141.179.34:3300/0,v1:220.141.179.34:6789/0] mon.host04; crush_location
{datacenter=DC2}
2: [v2:40.90.220.224:3300/0,v1:40.90.220.224:6789/0] mon.host01; crush_location
{datacenter=DC1}
3: [v2:60.140.141.144:3300/0,v1:60.140.141.144:6789/0] mon.host07; crush_location
{datacenter=DC1}
4: [v2:186.184.61.92:3300/0,v1:186.184.61.92:6789/0] mon.host03; crush_location
{datacenter=DC2}
dumped monmap epoch 19
```

8. Redeploy the monitors:

   Syntax

```
ceph orch apply mon --placement="HOST_1, HOST_2, HOST_3, HOST_4, HOST_5”
```

   Example

```
[ceph: root@host01 /]# ceph orch apply mon --placement="host01, host02, host04, host05,
host07"

Scheduled mon update...
```

9.2. REPLACING THE TIEBREAKER WITH A NEW MONITOR

If your tiebreaker monitor fails, you can replace it with a new monitor and remove it from the cluster.

Prerequisites

- A running Red Hat Ceph Storage cluster
- Stretch mode in enabled on a cluster

Procedure

1. Add a new monitor to the cluster:

   a. Manually add the **crush_location** to the new monitor:

      Syntax

      ```
      ceph mon add NEW_HOST IP_ADDRESS datacenter=DATACENTER
      ```

      Example

      ```
      [ceph: root@host01 /]# ceph mon add host06 213.222.226.50 datacenter=DC3

      adding mon.host06 at [v2:213.222.226.50:3300/0,v1:213.222.226.50:6789/0]
      ```

      NOTE

      The new monitor has to be in a different location than existing non-tiebreaker monitors.

   b. Disable automated monitor deployment:

      Example

      ```
      [ceph: root@host01 /]# ceph orch apply mon --unmanaged

      Scheduled mon update…
      ```

   c. Deploy the new monitor:

      Syntax

      ```
      ceph orch daemon add mon NEW_HOST
      ```

      Example

      ```
      [ceph: root@host01 /]# ceph orch daemon add mon host06
      ```

2. Ensure there are 6 monitors, from which 5 are in quorum:

   Example

   ```
   [ceph: root@host01 /]# ceph -s

   mon: 6 daemons, quorum host01, host02, host04, host05, host06 (age 30s), out of quorum: host07
   ```

3. Set the new monitor as a new tiebreaker:

   Syntax

```
ceph mon set_new_tiebreaker NEW_HOST
```

Example

```
[ceph: root@host01 /]# ceph mon set_new_tiebreaker host06
```

4. Remove the failed tiebreaker monitor:

Syntax

```
ceph orch daemon rm FAILED_TIEBREAKER_MONITOR --force
```

Example

```
[ceph: root@host01 /]# ceph orch daemon rm mon.host07 --force

Removed mon.host07 from host 'host07'
```

5. Verify that everything is configured properly:

Example

```
[ceph: root@host01 /]# ceph mon dump

epoch 19
fsid 1234ab78-1234-11ed-b1b1-de456ef0a89d
last_changed 2023-01-17T04:12:05.709475+0000
created 2023-01-16T05:47:25.631684+0000
min_mon_release 16 (pacific)
election_strategy: 3
stretch_mode_enabled 1
tiebreaker_mon host06
disallowed_leaders host06
0: [v2:213.222.226.50:3300/0,v1:213.222.226.50:6789/0] mon.host06; crush_location
{datacenter=DC3}
1: [v2:220.141.179.34:3300/0,v1:220.141.179.34:6789/0] mon.host04; crush_location
{datacenter=DC2}
2: [v2:40.90.220.224:3300/0,v1:40.90.220.224:6789/0] mon.host01; crush_location
{datacenter=DC1}
3: [v2:60.140.141.144:3300/0,v1:60.140.141.144:6789/0] mon.host02; crush_location
{datacenter=DC1}
4: [v2:186.184.61.92:3300/0,v1:186.184.61.92:6789/0] mon.host05; crush_location
{datacenter=DC2}
dumped monmap epoch 19
```

6. Redeploy the monitors:

Syntax

```
ceph orch apply mon --placement="HOST_1, HOST_2, HOST_3, HOST_4, HOST_5”
```

Example

-

```
[ceph: root@host01 /]# ceph orch apply mon --placement="host01, host02, host04, host05, host06"

Scheduled mon update…
```

9.3. FORCING STRETCH CLUSTER INTO RECOVERY OR HEALTHY MODE

When in stretch degraded mode, the cluster goes into the recovery mode automatically after the disconnected data center comes back. If that does not happen, or you want to enable recovery mode early, you can force the stretch cluster into the recovery mode.

Prerequisites

- A running Red Hat Ceph Storage cluster
- Stretch mode in enabled on a cluster

Procedure

1. Force the stretch cluster into the recovery mode:

   #### Example

   ```
   [ceph: root@host01 /]#  ceph osd force_recovery_stretch_mode --yes-i-really-mean-it
   ```

   NOTE

   The recovery state puts the cluster in the **HEALTH_WARN** state.

2. When in recovery mode, the cluster should go back into normal stretch mode after the placement groups are healthy. If that does not happen, you can force the stretch cluster into the healthy mode:

   #### Example

   ```
   [ceph: root@host01 /]#  ceph osd force_healthy_stretch_mode --yes-i-really-mean-it
   ```

   NOTE

   You can also run this command if you want to force the cross-data-center peering early and you are willing to risk data downtime, or you have verified separately that all the placement groups can peer, even if they are not fully recovered.

   You might also wish to invoke the healthy mode to remove the **HEALTH_WARN** state, which is generated by the recovery state.

NOTE

The **force_recovery_stretch_mode** and **force_recovery_healthy_mode** commands should not be necessary, as they are included in the process of managing unanticipated situations.

CHAPTER 10. 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.

Prerequisites

- Red Hat support account.

10.1. 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? 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.

10.2. 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.

Prerequisites

1. Install the debuginfo packages if they are not installed already.
   a. Enable the following repositories to install the required debuginfo packages.

   Example

   ```
   [root@host01 ~]# subscription-manager repos --enable=rhceph-6-tools-for-rhel-9-x86_64-rpms
   [root@host01 ~]# yum --enable=rhceph-6-tools-for-rhel-9-x86_64-debug-rpms
   ```

   Once the repository is enabled, you can install the debug info packages that you need from this list of supported packages:

```
ceph-base-debuginfo
ceph-common-debuginfo
ceph-debugsource
ceph-fuse-debuginfo
ceph-immutable-object-cache-debuginfo
ceph-mds-debuginfo
ceph-mgr-debuginfo
ceph-mon-debuginfo
ceph-osd-debuginfo
ceph-radosgw-debuginfo
cephfs-mirror-debuginfo
```

2. Ensure that the **gdb** package is installed and if it is not, install it:

Example

```
[root@host01 ~]# dnf install gdb
```

- Section 10.2.1, "Generating readable core dump files in containerized deployments"

10.2.1. Generating readable core dump files in containerized deployments

You can generate a core dump file for Red Hat Ceph Storage 6 which 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.
- Ensure the hosts has at least 8 GB RAM. If there are multiple daemons on the host, then Red Hat recommends more RAM.

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:

   Example

```
[root@mon]# echo "| /usr/lib/systemd/systemd-coredump %P %u %g %s %t %c %h %e"
> /proc/sys/kernel/core_pattern
```

b. Watch for the next container failure due to a Ceph process and search for the core dump file in the **/var/lib/systemd/coredump/** directory:

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 OSDs**:

a. Get the *MONITOR_ID* or the *OSD_ID* and enter the container:

Syntax

```
podman ps
podman exec -it MONITOR_ID_OR_OSD_ID bash
```

Example

```
[root@host01 ~]# podman ps
[root@host01 ~]# podman exec -it ceph-1ca9f6a8-d036-11ec-8263-fa163ee967ad-osd-2
bash
```

b. Install the **procps-ng** and **gdb** packages inside the container:

Example

```
[root@host01 ~]# dnf install procps-ng gdb
```

c. Find the process ID:

Syntax

```
ps -aef | grep PROCESS | grep -v run
```

Replace *PROCESS* with the name of the running process, for example **ceph-mon** or **ceph-osd**.

Example

```
[root@host01 ~]# 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
```

d. Generate the core dump file:

Syntax

```
gcore ID
```

Replace *ID* with the ID of the process that you got from the previous step, for example **18110**:

Example

```
[root@host01 ~]# 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.

Example

```
[root@host01 ~]# ls -ltr
total 709772
-rw-r--r--. 1 root root 726799544 Mar 18 19:46 core.18110
```

f. Copy the core dump file outside of the Ceph Monitor container:

Syntax

```
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.

3. To manually capture a core dump file for other Ceph daemons:

a. Log in to the **cephadm shell**:

Example

```
[root@host03 ~]# cephadm shell
```

b. Enable **ptrace** for the daemons:

Example

```
[ceph: root@host01 /]# ceph config set mgr mgr/cephadm/allow_ptrace true
```

c. Redeploy the daemon service:

Syntax

```
ceph orch redeploy SERVICE_ID
```

Example

```
[ceph: root@host01 /]# ceph orch redeploy mgr
[ceph: root@host01 /]# ceph orch redeploy rgw.rgw.1
```

d. Exit the **cephadm shell** and log in to the host where the daemons are deployed:

Example

```
[ceph: root@host01 /]# exit
[root@host01 ~]# ssh root@10.0.0.11
```

e. Get the *DAEMON_ID* and enter the container:

Example

```
[root@host04 ~]# podman ps
[root@host04 ~]# podman exec -it ceph-1ca9f6a8-d036-11ec-8263-fa163ee967ad-rgw-rgw-1-host04 bash
```

f. Install the **procps-ng** and **gdb** packages:

Example

```
[root@host04 /]# dnf install procps-ng gdb
```

g. Get the PID of process:

Example

```
[root@host04 /]# ps aux | grep rados
ceph           6  0.3  2.8 5334140 109052 ?      Sl   May10   5:25 /usr/bin/radosgw -n client.rgw.rgw.1.host04 -f --setuser ceph --setgroup ceph --default-log-to-file=false --default-log-to-stderr=true --default-log-stderr-prefix=debug
```

h. Gather core dump:

Syntax

```
gcore PID
```

Example

```
[root@host04 /]# gcore 6
```

i. Verify that the core dump file has been generated correctly.

Example

```
[root@host04 /]# ls -ltr
total 108798
-rw-r--r--. 1 root root 726799544 Mar 18 19:46 core.6
```

j. Copy the core dump file outside the container:

Syntax

```
podman cp ceph-mon-DAEMON_ID:/tmp/mon.core.PID /tmp
```

Replace *DAEMON_ID* with the ID number of the Ceph daemon and replace *PID* with the process ID number.

4. To allow **systemd-coredump** to successfully store the core dump for a crashed ceph daemon:

a. Set **DefaultLimitCORE** to *infinity* in **/etc/systemd/system.conf** to allow core dump collection for a crashed process:

Syntax

```
# cat /etc/systemd/system.conf

DefaultLimitCORE=infinity
```

b. Restart **systemd** or the node to apply the updated **systemd** settings:

Syntax

```
# sudo systemctl daemon-reexec
```

c. Verify the core dump files associated with previous daemon crashes:

Syntax

```
# ls -ltr /var/lib/systemd/coredump/
```

5. Upload the core dump file for analysis to a Red Hat support case. See *Providing information to Red Hat Support engineers* for details.

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

APPENDIX B. HEALTH MESSAGES OF A CEPH CLUSTER

There is a finite set of possible health messages that a Red Hat Ceph Storage cluster can raise. These are defined as health checks which have unique identifiers. The identifier is a terse pseudo-human-readable string that is intended to enable tools to make sense of health checks, and present them in a way that reflects their meaning.

Table B.1. Monitor

Health Code	Description
DAEMON_OLD_VERSION	Warn if old version of Ceph are running on any daemons. It will generate a health error if multiple versions are detected.
MON_DOWN	One or more Ceph Monitor daemons are currently down.
MON_CLOCK_SKEW	The clocks on the nodes running the **ceph-mon** daemons are not sufficiently well synchronized. Resolve it by synchronizing the clocks using **ntpd** or **chrony**.
MON_MSGR2_NOT_ENABLED	The **ms_bind_msgr2** option is enabled but one or more Ceph Monitors is not configured to bind to a v2 port in the cluster's monmap. Resolve this by running **ceph mon enable-msgr2** command.
MON_DISK_LOW	One or more Ceph Monitors are low on disk space.
MON_DISK_CRIT	One or more Ceph Monitors are critically low on disk space.
MON_DISK_BIG	The database size for one or more Ceph Monitors are very large.
AUTH_INSECURE_GLOBAL_ID_RECLAIM	One or more clients or daemons are connected to the storage cluster that are not securely reclaiming their **global_id** when reconnecting to a Ceph Monitor.
AUTH_INSECURE_GLOBAL_ID_RECLAIM_ALLOWED	Ceph is currently configured to allow clients to reconnect to monitors using an insecure process to reclaim their previous **global_id** because the setting **auth_allow_insecure_global_id_reclaim** is set to **true**.

Table B.2. Manager

Health Code	Description
MGR_DOWN	All Ceph Manager daemons are currently down.
MGR_MODULE_DEPENDENCY	An enabled Ceph Manager module is failing its dependency check.
MGR_MODULE_ERROR	A Ceph Manager module has experienced an unexpected error. Typically, this means an unhandled exception was raised from the module's serve function.

Table B.3. OSDs

Health Code	Description
OSD_DOWN	One or more OSDs are marked down.
OSD_CRUSH_TYPE_DOWN	All the OSDs within a particular CRUSH subtree are marked down, for example all OSDs on a host. For example, OSD_HOST_DOWN and OSD_ROOT_DOWN
OSD_ORPHAN	An OSD is referenced in the CRUSH map hierarchy but does not exist. Remove the OSD by running **ceph osd crush rm osd._OSD_ID** command.
OSD_OUT_OF_ORDER_FULL	The utilization thresholds for *nearfull*, *backfillfull*, *full*, or, *failsafefull* are not ascending. Adjust the thresholds by running **ceph osd set-nearfull-ratio *RATIO***, **ceph osd set-backfillfull-ratio *RATIO***, and **ceph osd set-full-ratio *RATIO***
OSD_FULL	One or more OSDs has exceeded the full threshold and is preventing the storage cluster from servicing writes. Restore write availability by raising the full threshold by a small margin **ceph osd set-full-ratio *RATIO***.
OSD_BACKFILLFULL	One or more OSDs has exceeded the backfillfull threshold, which will prevent data from being allowed to rebalance to this device.
OSD_NEARFULL	One or more OSDs has exceeded the nearfull threshold.

Health Code	Description
OSDMAP_FLAGS	One or more storage cluster flags of interest has been set. These flags include *full*, *pauserd*, *pausewr*, *noup*, *nodown*, *noin*, *noout*, *nobackfill*, *norecover*, *norebalance*, *noscrub*, *nodeep_scrub*, and *notieragent*. Except for *full*, the flags can be cleared with **ceph osd set *FLAG*** and **ceph osd unset *FLAG*** commands.
OSD_FLAGS	One or more OSDs or CRUSH has a flag of interest set. These flags include *noup*, *nodown*, *noin*, and *noout*.
OLD_CRUSH_TUNABLES	The CRUSH map is using very old settings and should be updated.
OLD_CRUSH_STRAW_CALC_VERSION	The CRUSH map is using an older, non-optimal method for calculating intermediate weight values for **straw** buckets.
CACHE_POOL_NO_HIT_SET	One or more cache pools is not configured with a hit set to track utilization, which will prevent the tiering agent from identifying cold objects to flush and evict from the cache. Configure the hit sets on the cache pool with **ceph osd pool set_POOL_NAME_ hit_set_type *TYPE***, **ceph osd pool set *POOL_NAME* hit_set_period *PERIOD_IN_SECONDS***, **ceph osd pool set *POOL_NAME* hit_set_count *NUMBER_OF_HIT_SETS***, and **ceph osd pool set *POOL_NAME* hit_set_fpp *TARGET_FALSE_POSITIVE_RATE*** commands.
OSD_NO_SORTBITWISE	**sortbitwise** flag is not set. Set the flag with **ceph osd set sortbitwise** command.
POOL_FULL	One or more pools has reached its quota and is no longer allowing writes. Increase the pool quota with **ceph osd pool set-quota *POOL_NAME* max_objects *NUMBER_OF_OBJECTS*** and **ceph osd pool set-quota *POOL_NAME* max_bytes *BYTES*** or delete some existing data to reduce utilization.
BLUEFS_SPILLOVER	One or more OSDs that use the BlueStore backend is allocated db partitions but that space has filled, such that metadata has "spilled over" onto the normal slow device. Disable this with **ceph config set osd bluestore_warn_on_bluefs_spillover false** command.

Health Code	Description
BLUEFS_AVAILABLE_SPACE	This output gives three values which are *BDEV_DB free*, *BDEV_SLOW free* and *available_from_bluestore*.
BLUEFS_LOW_SPACE	If the BlueStore File System (BlueFS) is running low on available free space and there is little **available_from_bluestore** one can consider reducing BlueFS allocation unit size.
BLUESTORE_FRAGMENTATION	As BlueStore works free space on underlying storage will get fragmented. This is normal and unavoidable but excessive fragmentation will cause slowdown.
BLUESTORE_LEGACY_STATFS	BlueStore tracks its internal usage statistics on a per-pool granular basis, and one or more OSDs have BlueStore volumes. Disable the warning with **ceph config set global bluestore_warn_on_legacy_statfs false** command.
BLUESTORE_NO_PER_POOL_OMAP	BlueStore tracks omap space utilization by pool. Disable the warning with **ceph config set global bluestore_warn_on_no_per_pool_omap false** command.
BLUESTORE_NO_PER_PG_OMAP	BlueStore tracks omap space utilization by PG. Disable the warning with **ceph config set global bluestore_warn_on_no_per_pg_omap false** command.
BLUESTORE_DISK_SIZE_MISMATCH	One or more OSDs using BlueStore has an internal inconsistency between the size of the physical device and the metadata tracking its size.
BLUESTORE_NO_COMPRESSION `	One or more OSDs is unable to load a BlueStore compression plugin. This can be caused by a broken installation, in which the **ceph-osd** binary does not match the compression plugins, or a recent upgrade that did not include a restart of the **ceph-osd** daemon.
BLUESTORE_SPURIOUS_READ_ERRORS	One or more OSDs using BlueStore detects spurious read errors at main device. BlueStore has recovered from these errors by retrying disk reads.

Table B.4. Device health

Health Code	Description
DEVICE_HEALTH	One or more devices is expected to fail soon, where the warning threshold is controlled by the **mgr/devicehealth/warn_threshold** config option. Mark the device *out* to migrate the data and replace the hardware.
DEVICE_HEALTH_IN_USE	One or more devices is expected to fail soon and has been marked "out" of the storage cluster based on **mgr/devicehealth/mark_out_threshold**, but it is still participating in one more PGs.
DEVICE_HEALTH_TOOMANY	Too many devices are expected to fail soon and the **mgr/devicehealth/self_heal** behavior is enabled, such that marking out all of the ailing devices would exceed the clusters **mon_osd_min_in_ratio** ratio that prevents too many OSDs from being automatically marked **out**.

Table B.5. Pools and placement groups

Health Code	Description
PG_AVAILABILITY	Data availability is reduced, meaning that the storage cluster is unable to service potential read or write requests for some data in the cluster.
PG_DEGRADED	Data redundancy is reduced for some data, meaning the storage cluster does not have the desired number of replicas for for replicated pools or erasure code fragments.
PG_RECOVERY_FULL	Data redundancy might be reduced or at risk for some data due to a lack of free space in the storage cluster, specifically, one or more PGs has the **recovery_toofull** flag set, which means that the cluster is unable to migrate or recover data because one or more OSDs is above the **full** threshold.
PG_BACKFILL_FULL	Data redundancy might be reduced or at risk for some data due to a lack of free space in the storage cluster, specifically, one or more PGs has the **backfill_toofull** flag set, which means that the cluster is unable to migrate or recover data because one or more OSDs is above the **backfillfull** threshold.

Health Code	Description
PG_DAMAGED	Data scrubbing has discovered some problems with data consistency in the storage cluster, specifically, one or more PGs has the inconsistent or **snaptrim_error** flag is set, indicating an earlier scrub operation found a problem, or that the **repair** flag is set, meaning a repair for such an inconsistency is currently in progress.
OSD_SCRUB_ERRORS	Recent OSD scrubs have uncovered inconsistencies.
OSD_TOO_MANY_REPAIRS	When a read error occurs and another replica is available it is used to repair the error immediately, so that the client can get the object data.
LARGE_OMAP_OBJECTS	One or more pools contain large omap objects as determined by **osd_deep_scrub_large_omap_object_key_threshold** or **osd_deep_scrub_large_omap_object_value_sum_threshold** or both. Adjust the thresholds with **ceph config set osd osd_deep_scrub_large_omap_object_key_threshold *KEYS*** and **ceph config set osd osd_deep_scrub_large_omap_object_value_sum_threshold *BYTES*** commands.
CACHE_POOL_NEAR_FULL	A cache tier pool is nearly full. Adjust the cache pool target size with **ceph osd pool set *CACHE_POOL_NAME* target_max_bytes *BYTES*** and **ceph osd pool set *CACHE_POOL_NAME* target_max_bytes *BYTES*** commands.
TOO_FEW_PGS	The number of PGs in use in the storage cluster is below the configurable threshold of **mon_pg_warn_min_per_osd** PGs per OSD.
POOL_PG_NUM_NOT_POWER_OF_TWO	One or more pools has a **pg_num** value that is not a power of two. Disable the warning with **ceph config set global mon_warn_on_pool_pg_num_not_power_of_two false** command.

Health Code	Description
POOL_TOO_FEW_PGS	One or more pools should probably have more PGs, based on the amount of data that is currently stored in the pool. You can either disable auto-scaling of PGs with **ceph osd pool set *POOL_NAME* pg_autoscale_mode off** command, automatically adjust the number of PGs with **ceph osd pool set *POOL_NAME* pg_autoscale_mode on** command or manually set the number of PGs with **ceph osd pool set *POOL_NAME* pg_num _NEW_PG_NUMBER** command.
TOO_MANY_PGS	The number of PGs in use in the storage cluster is above the configurable threshold of **mon_max_pg_per_osd** PGs per OSD. Increase the number of OSDs in the cluster by adding more hardware.
POOL_TOO_MANY_PGS	One or more pools should probably have more PGs, based on the amount of data that is currently stored in the pool. You can either disable auto-scaling of PGs with **ceph osd pool set *POOL_NAME* pg_autoscale_mode off** command, automatically adjust the number of PGs with **ceph osd pool set *POOL_NAME* pg_autoscale_mode on** command or manually set the number of PGs with **ceph osd pool set *POOL_NAME* pg_num _NEW_PG_NUMBER** command.
POOL_TARGET_SIZE_BYTES_OVERCOMMI TTED	One or more pools have a **target_size_bytes** property set to estimate the expected size of the pool, but the values exceed the total available storage. Set the value for the pool to zero with **ceph osd pool set *POOL_NAME* target_size_bytes 0** command.
POOL_HAS_TARGET_SIZE_BYTES_AND_RA TIO	One or more pools have both **target_size_bytes** and **target_size_ratio** set to estimate the expected size of the pool. Set the value for the pool to zero with **ceph osd pool set *POOL_NAME* target_size_bytes 0** command.
TOO_FEW_OSDS	The number of OSDs in the storage cluster is below the configurable threshold of **o`sd_pool_default_size**.

Health Code	Description
SMALLER_PGP_NUM	One or more pools has a **pgp_num** value less than **pg_num**. This is normally an indication that the PG count was increased without also increasing the placement behavior. Resolve this by setting **pgp_num** to match with **pg_num** with **ceph osd pool set *POOL_NAME* pgp_num *PG_NUM_VALUE*** command.
MANY_OBJECTS_PER_PG	One or more pools has an average number of objects per PG that is significantly higher than the overall storage cluster average. The specific threshold is controlled by the **mon_pg_warn_max_object_skew** configuration value.
POOL_APP_NOT_ENABLED	A pool exists that contains one or more objects but has not been tagged for use by a particular application. Resolve this warning by labeling the pool for use by an application with **rbd pool init *POOL_NAME*** command.
POOL_FULL	One or more pools has reached its quota. The threshold to trigger this error condition is controlled by the **mon_pool_quota_crit_threshold** configuration option.
POOL_NEAR_FULL	One or more pools is approaching a configured fullness threshold. Adjust the pool quotas with **ceph osd pool set-quota *POOL_NAME* max_objects *NUMBER_OF_OBJECTS*** and **ceph osd pool set-quota *POOL_NAME* max_bytes *BYTES*** commands.
OBJECT_MISPLACED	One or more objects in the storage cluster is not stored on the node the storage cluster would like it to be stored on. This is an indication that data migration due to some recent storage cluster change has not yet completed.
OBJECT_UNFOUND	One or more objects in the storage cluster cannot be found, specifically, the OSDs know that a new or updated copy of an object should exist, but a copy of that version of the object has not been found on OSDs that are currently online.
SLOW_OPS	One or more OSD or monitor requests is taking a long time to process. This can be an indication of extreme load, a slow storage device, or a software bug.

Health Code	Description
PG_NOT_SCRUBBED	One or more PGs has not been scrubbed recently. PGs are normally scrubbed within every configured interval specified by **osd_scrub_max_interval** globally. Initiate the scrub with **ceph pg scrub *PG_ID*** command.
PG_NOT_DEEP_SCRUBBED	One or more PGs has not been deep scrubbed recently. Initiate the scrub with **ceph pg deep-scrub *PG_ID*** command. PGs are normally scrubbed every **osd_deep_scrub_interval** seconds, and this warning triggers when **mon_warn_pg_not_deep_scrubbed_ratio** percentage of interval has elapsed without a scrub since it was due.
PG_SLOW_SNAP_TRIMMING	The snapshot trim queue for one or more PGs has exceeded the configured warning threshold. This indicates that either an extremely large number of snapshots were recently deleted, or that the OSDs are unable to trim snapshots quickly enough to keep up with the rate of new snapshot deletions.

Table B.6. Miscellaneous

Health Code	Description
RECENT_CRASH	One or more Ceph daemons has crashed recently, and the crash has not yet been acknowledged by the administrator.
TELEMETRY_CHANGED	Telemetry has been enabled, but the contents of the telemetry report have changed since that time, so telemetry reports will not be sent.
AUTH_BAD_CAPS	One or more auth users has capabilities that cannot be parsed by the monitor. Update the capabilities of the user with **ceph auth *ENTITY_NAME DAEMON_TYPE CAPS*** command.
OSD_NO_DOWN_OUT_INTERVAL	The **mon_osd_down_out_interval** option is set to zero, which means that the system will not automatically perform any repair or healing operations after an OSD fails. Silence the interval with **ceph config global mon mon_warn_on_osd_down_out_interval_zero false** command.

Health Code	Description
DASHBOARD_DEBUG	The Dashboard debug mode is enabled. This means, if there is an error while processing a REST API request, the HTTP error response contains a Python traceback. Disable the debug mode with **ceph dashboard debug disable** command.