Red Hat Ceph Storage 3

Configuration Guide

Configuration settings for Red Hat Ceph Storage

Last Updated: 2022-08-22

Red Hat Ceph Storage 3 Configuration Guide

Configuration settings for Red Hat Ceph Storage

Legal Notice

Abstract

This document provides instructions for configuring Red Hat Ceph Storage at boot time and run time. It also provides configuration reference information.

Table of Contents

CHAPTER 1. CONFIGURATION REFERENCE 4
1.1. GENERAL RECOMMENDATIONS 4
1.2. CONFIGURATION FILE STRUCTURE 4
1.3. METAVARIABLES 7
1.4. VIEWING THE CEPH RUNTIME CONFIGURATION 8
1.5. GETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME 8
1.6. SETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME 8
1.7. GENERAL CONFIGURATION REFERENCE 9
1.8. OSD MEMORY TARGET 10
1.9. MDS CACHE MEMORY LIMIT 11

CHAPTER 2. NETWORK CONFIGURATION REFERENCE 12
2.1. NETWORK CONFIGURATION SETTINGS 13
2.1.1. Public Network 13
2.1.2. Cluster Network 14
2.1.3. Verifying and configuring the MTU value 15
2.1.4. Messaging 16
2.1.5. AsyncMessenger Settings 18
2.1.6. Bind 20
2.1.7. Hosts 20
2.1.8. TCP 21
2.1.9. Firewall 22
2.1.9.1. Monitor Firewall 22
2.1.9.2. OSD Firewall 23
2.2. CEPH DAEMONS 23

CHAPTER 3. MONITOR CONFIGURATION REFERENCE 25
3.1. BACKGROUND 25
3.1.1. Cluster Maps 25
3.1.2. Monitor Quorum 26
3.1.3. Consistency 26
3.1.4. Bootstrapping Monitors 27
3.2. CONFIGURING MONITORS 27
3.2.1. Minimum Configuration 27
3.2.2. Cluster ID 28
3.2.3. Initial Members 29
3.2.4. Data 29
3.2.5. Storage Capacity 33
3.2.6. Heartbeat 34
3.2.7. Monitor Store Synchronization 35
3.2.8. Clock 41
3.2.9. Client 42
3.3. MISCELLANEOUS 43

CHAPTER 4. CEPHX CONFIGURATION REFERENCE 49
4.1. MANUAL 49
4.2. ENABLING AND DISABLING CEPHX 49
4.2.1. Enabling Cephx 49
4.2.2. Disabling Cephx 50
4.3. CONFIGURATION SETTINGS 51
4.3.1. Enablement 51
4.3.2. Keys 52

4.3.3. Daemon Keyrings 53
4.3.4. Signatures 54
4.3.5. Time to Live 55

CHAPTER 5. POOL, PG, AND CRUSH CONFIGURATION REFERENCE 56
5.1. SETTINGS 56

CHAPTER 6. OSD CONFIGURATION REFERENCE 62
6.1. GENERAL SETTINGS 62
6.2. JOURNAL SETTINGS 63
6.3. SCRUBBING 64
6.4. OPERATIONS 69
6.5. BACKFILLING 73
6.6. OSD MAP 74
6.7. RECOVERY 75
6.8. MISCELLANEOUS 77

CHAPTER 7. CONFIGURING MONITOR AND OSD INTERACTION 80
7.1. OSDS CHECK HEARTBEATS 80
7.2. OSDS REPORT DOWN OSDS 80
7.3. OSDS REPORT PEERING FAILURE 81
7.4. OSDS REPORT THEIR STATUS 82
7.5. CONFIGURATION SETTINGS 82
7.5.1. Monitor Settings 82
7.5.2. OSD Settings 85

CHAPTER 8. FILE STORE CONFIGURATION REFERENCE 88
8.1. EXTENDED ATTRIBUTES 88
8.2. SYNCHRONIZATION INTERVALS 91
8.3. FLUSHER 91
8.4. QUEUE 92
8.5. WRITEBACK THROTTLE 93
8.6. TIMEOUTS 96
8.7. B-TREE FILE SYSTEM 97
8.8. JOURNAL 97
8.9. MISCELLANEOUS 98

CHAPTER 9. JOURNAL CONFIGURATION REFERENCE 101
9.1. SETTINGS 101

CHAPTER 10. LOGGING CONFIGURATION REFERENCE 104
10.1. OSD 108
10.2. FILE STORE 109
10.3. THE CEPH OBJECT GATEWAY 109

CHAPTER 1. CONFIGURATION REFERENCE

All Ceph clusters have a configuration, which defines:

- Cluster identity
- Authentication settings
- Ceph daemon membership in the cluster
- Network configuration
- Host names and addresses
- Paths to keyrings
- Paths to data (including journals)
- Other runtime options

A deployment tool such as Red Hat Storage Console or Ansible will typically create an initial Ceph configuration file for you. However, you can create one yourself if you prefer to bootstrap a cluster without using a deployment tool.

For your convenience, each daemon has a series of default values, that is, many are set by the **ceph/src/common/config_opts.h** script. You can override these settings with a Ceph configuration file or at runtime by using the monitor **tell** command or connecting directly to a daemon socket on a Ceph node.

1.1. GENERAL RECOMMENDATIONS

You may maintain a Ceph configuration file anywhere you like, but Red Hat recommends having an administration node where you maintain a master copy of the Ceph configuration file.

When you make changes to the Ceph configuration file, it is a good practice to push the updated configuration file to your Ceph nodes to maintain consistency.

1.2. CONFIGURATION FILE STRUCTURE

The Ceph configuration file configures Ceph daemons at start time—overriding default values. Ceph configuration files use an *ini* style syntax. You can add comments by preceding comments with a pound sign (#) or a semi-colon (;). For example:

```
# <--A number (#) sign precedes a comment.
; A comment may be anything.
# Comments always follow a semi-colon (;) or a pound (#) on each line.
# The end of the line terminates a comment.
# We recommend that you provide comments in your configuration file(s).
```

The configuration file can configure all Ceph daemons in a Ceph storage cluster or all Ceph daemons of a particular type at start time. To configure a series of daemons, the settings must be included under the processes that will receive the configuration as follows:

[global]

Description

Settings under **[global]** affect all daemons in a Ceph Storage Cluster.

Example

auth supported = cephx

[osd]

Description

Settings under **[osd]** affect all **ceph-osd** daemons in the Ceph storage cluster, and override the same setting in **[global]**.

Example

osd journal size = 1000

[mon]

Description

Settings under **[mon]** affect all **ceph-mon** daemons in the Ceph storage cluster, and override the same setting in **[global]**.

Example

mon host = hostname1,hostname2,hostname3mon addr = 10.0.0.101:6789

[client]

Description

Settings under **[client]** affect all Ceph clients (for example, mounted Ceph block devices, Ceph object gateways, and so on).

Example

log file = /var/log/ceph/radosgw.log

Global settings affect all instances of all daemon in the Ceph storage cluster. Use the **[global]** setting for values that are common for all daemons in the Ceph storage cluster. You can override each **[global]** setting by:

1. Changing the setting in a particular process type (for example, **[osd]**, **[mon]**).
2. Changing the setting in a particular process (for example, **[osd.1]**).

Overriding a global setting affects all child processes, except those that you specifically override in a particular daemon.

A typical global setting involves activating authentication. For example:

```
[global]
#Enable authentication between hosts within the cluster.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

You can specify settings that apply to a particular type of daemon. When you specify settings under **[osd]** or **[mon]** without specifying a particular instance, the setting will apply to all OSD or monitor daemons respectively.

A typical daemon-wide setting involves setting journal sizes, filestore settings, and so on For example:

```
[osd]
osd_journal_size = 1000
```

You can specify settings for particular instances of a daemon. You may specify an instance by entering its type, delimited by a period (.) and by the instance ID. The instance ID for a Ceph OSD daemons is always numeric, but it may be alphanumeric for Ceph monitors.

```
[osd.1]
# settings affect osd.1 only.

[mon.a]
# settings affect mon.a only.
```

The default Ceph configuration file locations in sequential order include:

1. **$CEPH_CONF** (the path following the **$CEPH_CONF** environment variable)
2. **-c path/path** (the **-c** command line argument)
3. **/etc/ceph/ceph.conf**
4. **~/.ceph/config**
5. **./ceph.conf** (in the current working directory)

A typical Ceph configuration file has at least the following settings:

```
[global]
fsid = {cluster-id}
mon_initial_members = {hostname}[, {hostname}]
mon_host = {ip-address}[, {ip-address}]

#All clusters have a front-side public network.
#If you have two NICs, you can configure a back side cluster
#network for OSD object replication, heart beats, backfilling,
#recovery, and so on
public_network = {network}[, {network}]
#cluster_network = {network}[, {network}]

#Clusters require authentication by default.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx

#Choose reasonable numbers for your journals, number of replicas
#and placement groups.
osd_journal_size = {n}
osd_pool_default_size = {n}  # Write an object n times.
osd_pool_default_min_size = {n} # Allow writing n copy in a degraded state.
osd_pool_default_pg_num = {n}
osd_pool_default_pgp_num = {n}

#Choose a reasonable crush leaf type.
#0 for a 1-node cluster.
#1 for a multi node cluster in a single rack
```

```
#2 for a multi node, multi chassis cluster with multiple hosts in a chassis
#3 for a multi node cluster with hosts across racks, and so on
osd_crush_chooseleaf_type = {n}
```

1.3. METAVARIABLES

Metavariables simplify Ceph storage cluster configuration dramatically. When a metavariable is set in a configuration value, Ceph expands the metavariable into a concrete value.

Metavariables are very powerful when used within the **[global]**, **[osd]**, **[mon]**, or **[client]** sections of the Ceph configuration file. However, you can also use them with the administration socket. Ceph metavariables are similar to Bash shell expansion.

Ceph supports the following metavariables:

$cluster

Description

Expands to the Ceph storage cluster name. Useful when running multiple Ceph storage clusters on the same hardware.

Example

/etc/ceph/$cluster.keyring

Default

ceph

$type

Description

Expands to one of **osd** or **mon**, depending on the type of the instant daemon.

Example

/var/lib/ceph/$type

$id

Description

Expands to the daemon identifier. For **osd.0**, this would be **0**.

Example

/var/lib/ceph/$type/$cluster-$id

$host

Description

Expands to the host name of the instant daemon.

$name

Description

Expands to **$type.$id**.

Example

/var/run/ceph/$cluster-$name.asok

1.4. VIEWING THE CEPH RUNTIME CONFIGURATION

To view a runtime configuration, log in to a Ceph node and execute:

```
ceph daemon {daemon-type}.{id} config show
```

For example, if you want to see the configuration for **osd.0**, log into the node containing **osd.0** and execute:

```
ceph daemon osd.0 config show
```

For additional options, specify a daemon and **help**. For example:

```
ceph daemon osd.0 help
```

1.5. GETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME

To get a specific configuration setting at runtime, log in to a Ceph node and execute:

```
ceph daemon {daemon-type}.{id} config get {parameter}
```

For example to retrieve the public address of **osd.0**, execute:

```
ceph daemon osd.0 config get public_addr
```

1.6. SETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME

There are two general ways to set a runtime configuration:

- by using the Ceph monitor
- by using the administration socket

You can set a Ceph runtime configuration setting by contacting the monitor using the **tell** and **injectargs** command. To use this approach, the monitors and the daemon you are trying to modify must be running:

```
ceph tell {daemon-type}.{daemon id or *} injectargs --{name} {value} [--{name} {value}]
```

Replace **{daemon-type}** with one of **osd** or **mon**. You can apply the runtime setting to all daemons of a particular type with *, or specify a specific daemon's ID (that is, its number or name). For example, to change the debug logging for a **ceph-osd** daemon named **osd.0** to **0/5**, execute the following command:

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

The **tell** command takes multiple arguments, so each argument for **tell** must be within single quotes, and the configuration prepended with two dashes (**'--{config_opt} {opt-val}' ['-{config_opt} {opt-val}']**). Quotes are not necessary for the **daemon** command, because it only takes one argument.

The **ceph tell** command goes through the monitors. If you cannot bind to the monitor, you can still make the change by logging into the host of the daemon whose configuration you want to change using **ceph daemon**. For example:

```
sudo ceph osd.0 config set debug_osd 0/5
```

1.7. GENERAL CONFIGURATION REFERENCE

General settings typically get set automatically by deployment tools.

fsid

Description

The file system ID. One per cluster.

Type

UUID

Required

No.

Default

N/A. Usually generated by deployment tools.

admin_socket

Description

The socket for executing administrative commands on a daemon, irrespective of whether Ceph monitors have established a quorum.

Type

String

Required

No

Default

/var/run/ceph/$cluster-$name.asok

pid_file

Description

The file in which the monitor or OSD will write its PID. For instance, **/var/run/$cluster/$type.$id.pid** will create /var/run/ceph/mon.a.pid for the **mon** with id **a** running in the **ceph** cluster. The **pid file** is removed when the daemon stops gracefully. If the process is not daemonized (meaning it runs with the **-f** or **-d** option), the **pid file** is not created.

Type

String

Required

No

Default

No

chdir

Description

The directory Ceph daemons change to once they are up and running. Default / directory recommended.

Type

String

Required

No

Default

/

max_open_files

Description

If set, when the Red Hat Ceph Storage cluster starts, Ceph sets the **max_open_fds** at the OS level (that is, the max # of file descriptors). It helps prevents Ceph OSDs from running out of file descriptors.

Type

64-bit Integer

Required

No

Default

0

fatal_signal_handlers

Description

If set, we will install signal handlers for SEGV, ABRT, BUS, ILL, FPE, XCPU, XFSZ, SYS signals to generate a useful log message.

Type

Boolean

Default

true

1.8. OSD MEMORY TARGET

BlueStore keeps OSD heap memory usage under a designated target size with the **osd_memory_target** configuration option.

The option **osd_memory_target** sets OSD memory based upon the available RAM in the system. By default, Anisble sets the value to 4 GB. You can change the value, expressed in bytes, in the **/usr/share/ceph-ansible/group_vars/all.yml** file when deploying the daemon.

Example: Set the **osd_memory_target** to 6000000000 bytes

```
ceph_conf_overrides:
  osd:
    osd_memory_target=6000000000
```

Ceph OSD memory caching is more important when the block device is slow, for example, traditional hard drives, because the benefit of a cache hit is much higher than it would be with a solid state drive. However, this has to be weighed-in to co-locate OSDs with other services, such as in a hyper-converged infrastructure (HCI), or other applications.

NOTE

The value of **osd_memory_target** is one OSD per device for traditional hard drive device, and two OSDs per device for NVMe SSD devices. The **osds_per_device** is defined in **group_vars/osds.yml** file.

Additional Resources

- Setting **osd_memory_target** Setting OSD Memory Target

1.9. MDS CACHE MEMORY LIMIT

MDS servers keep their metadata in a separate storage pool, named **cephfs_metadata**, and are the users of Ceph OSDs. For Ceph File Systems, MDS servers have to support an entire Red Hat Ceph Storage cluster, not just a single storage device within the storage cluster, so their memory requirements can be significant, particularly if the workload consists of small-to-medium-size files, where the ratio of metadata to data is much higher.

Example:Set the **mds_cache_memory_limit** to 2000000000 bytes

```
ceph_conf_overrides:
  osd:
    mds_cache_memory_limit=2000000000
```

NOTE

For a large Red Hat Ceph Storage cluster with a metadata-intensive workload, do not put an MDS server on the same node as other memory-intensive services, doing so gives you the option to allocate more memory to MDS, for example, sizes greater than 100 GB.

Additional Resources

- See Understanding MDS Cache Size Limits

CHAPTER 2. NETWORK CONFIGURATION REFERENCE

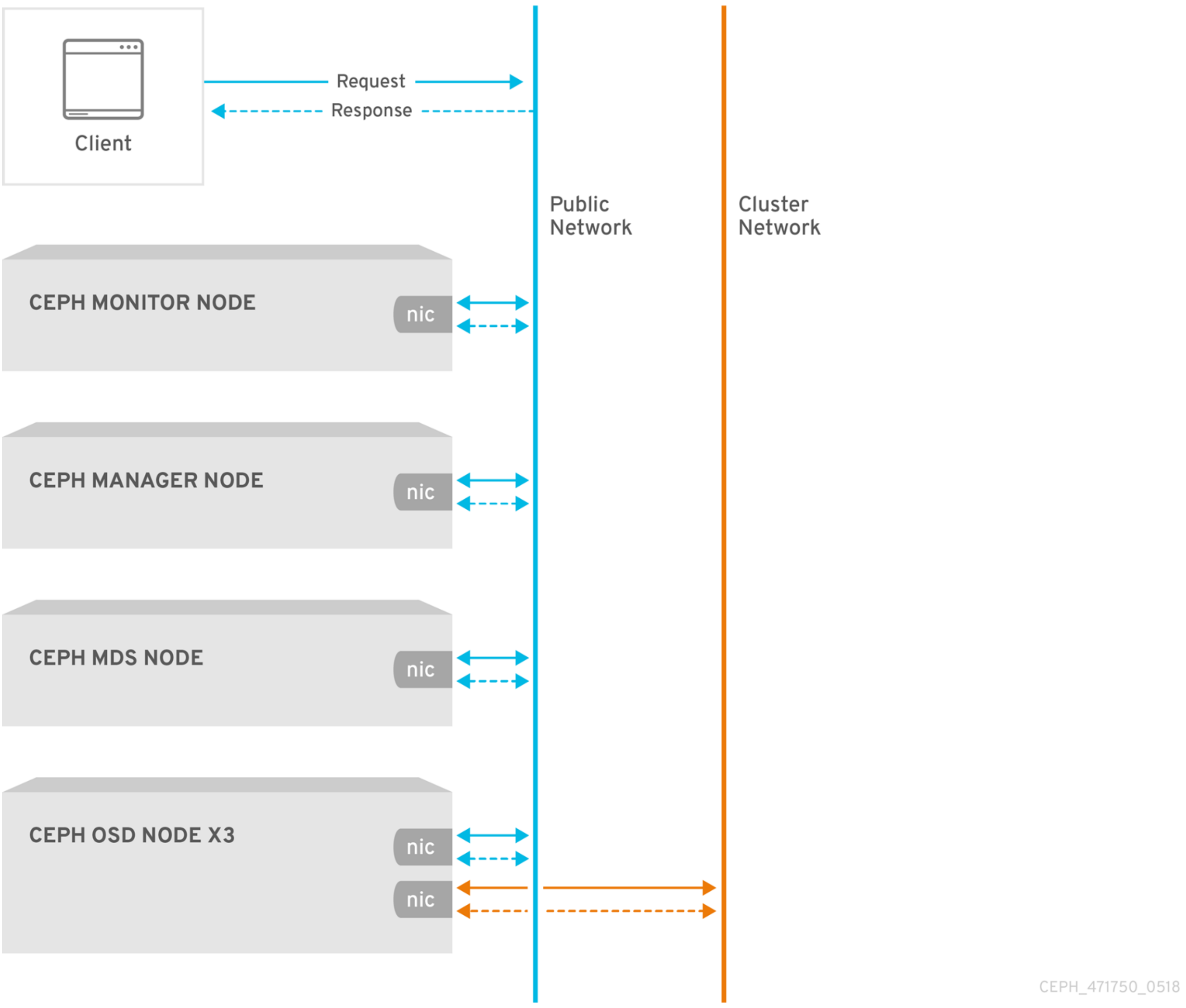
Network configuration is critical for building a high performance Red Hat Ceph Storage cluster. The Ceph storage cluster does not perform request routing or dispatching on behalf of the Ceph client. Instead, Ceph clients make requests directly to Ceph OSD daemons. Ceph OSDs perform data replication on behalf of Ceph clients, which means replication and other factors impose additional loads on the networks of Ceph storage clusters.

All Ceph clusters must use a public network. However, unless you specify a cluster (internal) network, Ceph assumes a single public network. Ceph can function with a public network only, but you will see significant performance improvement with a second "cluster" network in a large cluster.

Red Hat recommends running a Ceph storage cluster with two networks:

- a public network
- and a cluster network.

To support two networks, each Ceph Node will need to have more than one network interface card (NIC).



There are several reasons to consider operating two separate networks:

- **Performance:** Ceph OSDs handle data replication for the Ceph clients. When Ceph OSDs replicate data more than once, the network load between Ceph OSDs easily dwarfs the network load between Ceph clients and the Ceph storage cluster. This can introduce latency and create a performance problem. Recovery and rebalancing can also introduce significant latency on the public network.

- **Security**: While most people are generally civil, some actors will engage in what is known as a Denial of Service (DoS) attack. When traffic between Ceph OSDs gets disrupted, peering may fail and placement groups may no longer reflect an **active + clean** state, which may prevent users from reading and writing data. A great way to defeat this type of attack is to maintain a completely separate cluster network that does not connect directly to the internet.

2.1. NETWORK CONFIGURATION SETTINGS

Network configuration settings are not required. Ceph can function with a public network only, assuming a public network is configured on all hosts running a Ceph daemon. However, Ceph allows you to establish much more specific criteria, including multiple IP networks and subnet masks for your public network. You can also establish a separate cluster network to handle OSD heartbeat, object replication, and recovery traffic.

Do not confuse the IP addresses you set in the configuration with the public-facing IP addresses network clients might use to access your service. Typical internal IP networks are often **192.168.0.0** or **10.0.0.0**.

TIP

If you specify more than one IP address and subnet mask for either the public or the cluster network, the subnets within the network must be capable of routing to each other. Additionally, make sure you include each IP address/subnet in your IP tables and open ports for them as necessary.

NOTE

Ceph uses CIDR notation for subnets (for example, **10.0.0.0/24**).

When you configured the networks, you can restart the cluster or restart each daemon. Ceph daemons bind dynamically, so you do not have to restart the entire cluster at once if you change the network configuration.

2.1.1. Public Network

To configure a public network, add the following option to the **[global]** section of the Ceph configuration file.

```
[global]
   ...
   public_network = <public-network/netmask>
```

The public network configuration allows you specifically define IP addresses and subnets for the public network. You may specifically assign static IP addresses or override **public network** settings using the **public addr** setting for a specific daemon.

public_network

Description

The IP address and netmask of the public (front-side) network (for example, **192.168.0.0/24**). Set in **[global]**. You can specify comma-delimited subnets.

Type

<ip-address>/<netmask> [, <ip-address>/<netmask>]

Required

No

Default

N/A

public_addr

Description

The IP address for the public (front-side) network. Set for each daemon.

Type

IP Address

Required

No

Default

N/A

2.1.2. Cluster Network

If you declare a cluster network, OSDs will route heartbeat, object replication, and recovery traffic over the cluster network. This can improve performance compared to using a single network. To configure a cluster network, add the following option to the **[global]** section of the Ceph configuration file.

```
[global]
    ...
    cluster_network = <cluster-network/netmask>
```

It is preferable, that the cluster network is not reachable from the public network or the Internet for added security.

The cluster network configuration allows you to declare a cluster network, and specifically define IP addresses and subnets for the cluster network. You can specifically assign static IP addresses or override **cluster network** settings using the **cluster addr** setting for specific OSD daemons.

cluster_network

Description

The IP address and netmask of the cluster network (for example, **10.0.0.0/24**). Set in **[global]**. You can specify comma-delimited subnets.

Type

<ip-address>/<netmask> [, <ip-address>/<netmask>]

Required

No

Default

N/A

cluster_addr

Description

The IP address for the cluster network. Set for each daemon.

Type

Address

Required

No

Default

N/A

2.1.3. Verifying and configuring the MTU value

The maximum transmission unit (MTU) value is the size, in bytes, of the largest packet sent on the link layer. The default MTU value is 1500 bytes. Red Hat recommends using jumbo frames, a MTU value of 9000 bytes, for a Red Hat Ceph Storage cluster.

IMPORTANT

Red Hat Ceph Storage requires the same MTU value throughout all networking devices in the communication path, end-to-end for both public and cluster networks. Verify that the MTU value is the same on all nodes and networking equipment in the environment before using a Red Hat Ceph Storage cluster in production.

NOTE

When bonding network interfaces together, the MTU value only needs to be set on the bonded interface. The new MTU value propagates from the bonding device to the underlying network devices.

Prerequisites

- Root-level access to the node.

Procedure

1. Verify the current MTU value:

   Example

   ```
   [root@mon ~]# ip link list
   1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
   DEFAULT group default qlen 1000
       link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   2: enp22s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
   mode DEFAULT group default qlen 1000
   ```

   For this example, the network interface is **enp22s0f0** and it has a MTU value of **1500**.

2. To **temporarily** change the MTU value online:

Syntax

```
ip link set dev NET_INTERFACE mtu NEW_MTU_VALUE
```

Example

```
[root@mon ~]# ip link set dev enp22s0f0 mtu 9000
```

3. To **permanently** change the MTU value.

   a. Open for editing the network configuration file for that particular network interface:

   Syntax

   ```
   vim /etc/sysconfig/network-scripts/ifcfg-NET_INTERFACE
   ```

   Example

   ```
   [root@mon ~]# vim /etc/sysconfig/network-scripts/ifcfg-enp22s0f0
   ```

   b. On a new line, add the **MTU=9000** option:

   Example

   ```
   NAME="enp22s0f0"
   DEVICE="enp22s0f0"
   MTU=9000 1
   ONBOOT=yes
   NETBOOT=yes
   UUID="a8c1f1e5-bd62-48ef-9f29-416a102581b2"
   IPV6INIT=yes
   BOOTPROTO=dhcp
   TYPE=Ethernet
   ```

   c. Restart the network service:

   Example

   ```
   [root@mon ~]# systemctl restart network
   ```

Additional Resources

- For more details, see the *Networking Guide* for Red Hat Enterprise Linux 7.

2.1.4. Messaging

Messenger is the Ceph network layer implementation. Red Hat supports two messenger types:

- **simple**
- **async**

In RHCS 2 and earlier releases, **simple** is the default messenger type. In RHCS 3, **async** is the default messenger type. To change the messenger type, specify the **ms_type** configuration setting in the **[global]** section of the Ceph configuration file.

NOTE

For the **async** messenger, Red Hat supports the **posix** transport type, but does not currently support **rdma** or **dpdk**. By default, the **ms_type** setting in RHCS 3 should reflect **async+posix**, where **async** is the messenger type and **posix** is the transport type.

About SimpleMessenger

The **SimpleMessenger** implementation uses TCP sockets with two threads per socket. Ceph associates each logical session with a connection. A pipe handles the connection, including the input and output of each message. While **SimpleMessenger** is effective for the **posix** transport type, it is not effective for other transport types such as **rdma** or **dpdk**. Consequently, **AsyncMessenger** is the default messenger type for RHCS 3 and later releases.

About AsyncMessenger

For RHCS 3, the **AsyncMessenger** implementation uses TCP sockets with a fixed-size thread pool for connections, which should be equal to the highest number of replicas or erasure-code chunks. The thread count can be set to a lower value if performance degrades due to a low CPU count or a high number of OSDs per server.

NOTE

Red Hat does not support other transport types such as **rdma** or **dpdk** at this time.

Messenger Type Settings

ms_type

Description

The messenger type for the network transport layer. Red Hat supports the **simple** and the **async** messenger type using **posix** semantics.

Type

String.

Required

No.

Default

async+posix

ms_public_type

Description

The messenger type for the network transport layer of the public network. It operates identically to **ms_type**, but is applicable only to the public or front-side network. This setting enables Ceph to use a different messenger type for the public or front-side and cluster or back-side networks.

Type

String.

Required

No.

Default

None.

ms_cluster_type

Description

The messenger type for the network transport layer of the cluster network. It operates identically to **ms_type**, but is applicable only to the cluster or back-side network. This setting enables Ceph to use a different messenger type for the public or front-side and cluster or back-side networks.

Type

String.

Required

No.

Default

None.

2.1.5. AsyncMessenger Settings

ms_async_transport_type

Description

Transport type used by the **AsyncMessenger**. Red Hat supports the **posix** setting, but does not support the **dpdk** or **rdma** settings at this time. POSIX uses standard TCP/IP networking and is the default value. Other transport types are experimental and are **NOT** supported.

Type

String

Required

No

Default

posix

ms_async_op_threads

Description

Initial number of worker threads used by each **AsyncMessenger** instance. This configuration setting **SHOULD** equal the number of replicas or erasure code chunks. but it may be set lower if the CPU core count is low or the number of OSDs on a single server is high.

Type

64-bit Unsigned Integer

Required

No

Default

3

ms_async_max_op_threads

Description

The maximum number of worker threads used by each **AsyncMessenger** instance. Set to lower values if the OSD host has limited CPU count, and increase if Ceph is underutilizing CPUs are underutilized.

Type

64-bit Unsigned Integer

Required

No

Default

5

ms_async_set_affinity

Description

Set to **true** to bind **AsyncMessenger** workers to particular CPU cores.

Type

Boolean

Required

No

Default

true

ms_async_affinity_cores

Description

When **ms_async_set_affinity** is **true**, this string specifies how **AsyncMessenger** workers are bound to CPU cores. For example, **0,2** will bind workers #1 and #2 to CPU cores #0 and #2, respectively. **NOTE:** When manually setting affinity, make sure to not assign workers to virtual CPUs created as an effect of hyper threading or similar technology, because they are slower than physical CPU cores.

Type

String

Required

No

Default

(empty)

ms_async_send_inline

Description

Send messages directly from the thread that generated them instead of queuing and sending from the **AsyncMessenger** thread. This option is known to decrease performance on systems with a lot of CPU cores, so it’s disabled by default.

Type

Boolean

Required

No

Default

false

2.1.6. Bind

Bind settings set the default port ranges Ceph OSD daemons use. The default range is **6800:7100**. Ensure that the firewall configuration allows you to use the configured port range.

You can also enable Ceph daemons to bind to IPv6 addresses.

ms_bind_port_min

Description

The minimum port number to which an OSD daemon will bind.

Type

32-bit Integer

Default

6800

Required

No

ms_bind_port_max

Description

The maximum port number to which an OSD daemon will bind.

Type

32-bit Integer

Default

7300

Required

No.

ms_bind_ipv6

Description

Enables Ceph daemons to bind to IPv6 addresses.

Type

Boolean

Default

false

Required

No

2.1.7. Hosts

Ceph expects at least one monitor declared in the Ceph configuration file, with a **mon addr** setting under each declared monitor. Ceph expects a **host** setting under each declared monitor, metadata server and OSD in the Ceph configuration file.

mon_addr

Description

A list of **<hostname>:<port>** entries that clients can use to connect to a Ceph monitor. If not set, Ceph searches **[mon.*]** sections.

Type

String

Required

No

Default

N/A

host

Description

The host name. Use this setting for specific daemon instances (for example, **[osd.0]**).

Type

String

Required

Yes, for daemon instances.

Default

localhost

TIP

Do not use **localhost**. To get your host name, execute the **hostname -s** command and use the name of your host to the first period, not the fully-qualified domain name.

IMPORTANT

Do not specify any value for **host** when using a third party deployment system that retrieves the host name for you.

2.1.8. TCP

Ceph disables TCP buffering by default.

ms_tcp_nodelay

Description

Ceph enables **ms_tcp_nodelay** so that each request is sent immediately (no buffering). Disabling Nagle’s algorithm increases network traffic, which can introduce congestion. If you experience large numbers of small packets, you may try disabling **ms_tcp_nodelay**, but be aware that disabling it will generally increase latency.

Type

Boolean

Required

No

Default

true

ms_tcp_rcvbuf

Description

The size of the socket buffer on the receiving end of a network connection. Disable by default.

Type

32-bit Integer

Required

No

Default

0

ms_tcp_read_timeout

Description

If a client or daemon makes a request to another Ceph daemon and does not drop an unused connection, the **tcp read timeout** defines the connection as idle after the specified number of seconds.

Type

Unsigned 64-bit Integer

Required

No

Default

900 15 minutes.

2.1.9. Firewall

By default, daemons bind to ports within the **6800:7100** range. You can configure this range at your discretion. Before configuring the firewall, check the default firewall configuration. You can configure this range at your discretion.

```
sudo iptables -L
```

For the **firewalld** daemon, execute the following command as **root**:

```
# firewall-cmd --list-all-zones
```

Some Linux distributions include rules that reject all inbound requests except SSH from all network interfaces. For example:

```
REJECT all -- anywhere anywhere reject-with icmp-host-prohibited
```

2.1.9.1. Monitor Firewall

Ceph monitors listen on port **6789** by default. Additionally, Ceph monitors always operate on the public network. When you add the rule using the example below, make sure you replace **<iface>** with the public network interface (for example, **eth0**, **eth1**, and so on), **<ip-address>** with the IP address of the public network and **<netmask>** with the netmask for the public network.

```
sudo iptables -A INPUT -i <iface> -p tcp -s <ip-address>/<netmask> --dport 6789 -j ACCEPT
```

_

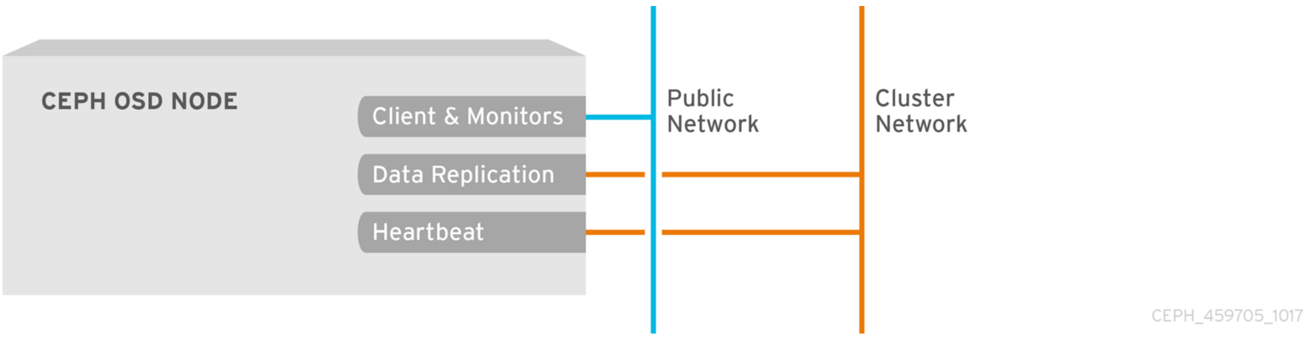
For the **firewalld** daemon, execute the following commands as **root**:

```
# firewall-cmd --zone=public --add-port=6789/tcp
# firewall-cmd --zone=public --add-port=6789/tcp --permanent
```

2.1.9.2. OSD Firewall

By default, Ceph OSDs bind to the first available ports on a Ceph node beginning at port 6800. Ensure to open at least three ports beginning at port 6800 for each OSD that runs on the host:

1. One for talking to clients and monitors (public network).
2. One for sending data to other OSDs (cluster network).
3. One for sending heartbeat packets (cluster network).



Ports are node-specific. However, you might need to open more ports than the number of ports needed by Ceph daemons running on that Ceph node in the event that processes get restarted and the bound ports do not get released. Consider to open a few additional ports in case a daemon fails and restarts without releasing the port such that the restarted daemon binds to a new port. Also, consider opening the port range of **6800:7300** on each OSD host.

If you set separate public and cluster networks, you must add rules for both the public network and the cluster network, because clients will connect using the public network and other Ceph OSD Daemons will connect using the cluster network.

When you add the rule using the example below, make sure you replace **<iface>** with the network interface (for example, **eth0** or **eth1), `<ip-address>** with the IP address and **<netmask>** with the netmask of the public or cluster network. For example:

```
sudo iptables -A INPUT -i <iface>  -m multiport -p tcp -s <ip-address>/<netmask> --dports 6800:6810
-j ACCEPT
```

For the **firewalld** daemon, execute the following commands as **root**:

```
# firewall-cmd --zone=public --add-port=6800-6810/tcp
# firewall-cmd --zone=public --add-port=6800-6810/tcp --permanent
```

If you put the cluster network into another zone, open the ports within that zone as appropriate.

2.2. CEPH DAEMONS

Ceph has one network configuration requirement that applies to all daemons. The Ceph configuration file must specify the **host** for each daemon. Ceph no longer requires that a Ceph configuration file specify the monitor IP address and its port.

IMPORTANT

Some deployment utilities might create a configuration file for you. Do not set these values if the deployment utility does it for you.

TIP

The **host** setting is the short name of the host (that is, not an FQDN). It is not an IP address either. Use the **hostname -s** command to retrieve the name of the host.

```
[mon.a]

    host = <hostname>
    mon addr = <ip-address>:6789

[osd.0]
    host = <hostname>
```

You do not have to set the host IP address for a daemon. If you have a static IP configuration and both public and cluster networks running, the Ceph configuration file might specify the IP address of the host for each daemon. To set a static IP address for a daemon, the following option(s) should appear in the daemon instance sections of the Ceph configuration file.

```
[osd.0]
    public_addr = <host-public-ip-address>
    cluster_addr = <host-cluster-ip-address>
```

One NIC OSD in a Two Network Cluster

Generally, Red Hat does not recommend deploying an OSD host with a single NIC in a cluster with two networks. However, you cam accomplish this by forcing the OSD host to operate on the public network by adding a **public addr** entry to the **[osd.n]** section of the Ceph configuration file, where **n** refers to the number of the OSD with one NIC. Additionally, the public network and cluster network must be able to route traffic to each other, which Red Hat does not recommend for security reasons.

CHAPTER 3. MONITOR CONFIGURATION REFERENCE

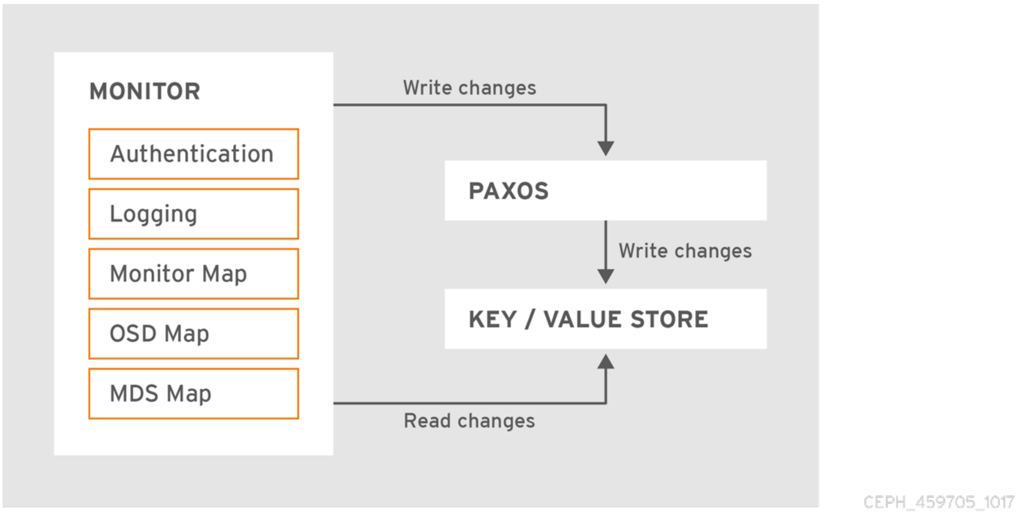
Understanding how to configure a Ceph monitor is an important part of building a reliable Red Hat Ceph Storage cluster. All clusters have at least one monitor. A monitor configuration usually remains fairly consistent, but you can add, remove or replace a monitor in a cluster.

3.1. BACKGROUND

Ceph monitors maintain a "master copy" of the cluster map. That means a Ceph client can determine the location of all Ceph monitors and Ceph OSDs just by connecting to one Ceph monitor and retrieving a current cluster map.

Before Ceph clients can read from or write to Ceph OSDs, they must connect to a Ceph monitor first. With a current copy of the cluster map and the CRUSH algorithm, a Ceph client can compute the location for any object. The ability to compute object locations allows a Ceph client to talk directly to Ceph OSDs, which is a very important aspect of Ceph high scalability and performance.

The primary role of the Ceph monitor is to maintain a master copy of the cluster map. Ceph monitors also provide authentication and logging services. Ceph monitors write all changes in the monitor services to a single Paxos instance, and Paxos writes the changes to a key-value store for strong consistency. Ceph monitors can query the most recent version of the cluster map during synchronization operations. Ceph monitors leverage the key-value store's snapshots and iterators (using the **leveldb** database) to perform store-wide synchronization.



3.1.1. Cluster Maps

The cluster map is a composite of maps, including the monitor map, the OSD map, and the placement group map. The cluster map tracks a number of important events:

- Which processes are **in** the Red Hat Ceph Storage cluster
- Which processes that are **in** the Red Hat Ceph Storage cluster are **up** and running or **down**.
- Whether, the placement groups are **active** or **inactive**, and **clean** or in some other state.
- other details that reflect the current state of the cluster such as:
  - the total amount of storage space or
  - the amount of storage used.

When there is a significant change in the state of the cluster for example, a Ceph OSD goes down, a placement group falls into a degraded state, and so on, the cluster map gets updated to reflect the current state of the cluster. Additionally, the Ceph monitor also maintains a history of the prior states of the cluster. The monitor map, OSD map, and placement group map each maintain a history of their map versions. Each version is called an **epoch**.

When operating the Red Hat Ceph Storage cluster, keeping track of these states is an important part of the cluster administration.

3.1.2. Monitor Quorum

A cluster will run sufficiently with a single monitor. However, a single monitor is a single-point-of-failure. To ensure high availability in a production Ceph storage cluster, run Ceph with multiple monitors so that the failure of a single monitor will not cause a failure of the entire cluster.

When a Ceph storage cluster runs multiple Ceph monitors for high availability, Ceph monitors use the Paxos algorithm to establish consensus about the master cluster map. A consensus requires a majority of monitors running to establish a quorum for consensus about the cluster map (for example, 1; 2 out of 3; 3 out of 5; 4 out of 6; and so on).

mon_force_quorum_join

Description

Force monitor to join quorum even if it has been previously removed from the map

Type

Boolean

Default

False

3.1.3. Consistency

When you add monitor settings to the Ceph configuration file, you need to be aware of some of the architectural aspects of Ceph monitors. Ceph imposes strict consistency requirements for a Ceph monitor when discovering another Ceph monitor within the cluster. Whereas, Ceph clients and other Ceph daemons use the Ceph configuration file to discover monitors, monitors discover each other using the monitor map (**monmap**), not the Ceph configuration file.

A Ceph monitor always refers to the local copy of the monitor map when discovering other Ceph monitors in the Red Hat Ceph Storage cluster. Using the monitor map instead of the Ceph configuration file avoids errors that could break the cluster, for example, typos in the Ceph configuration file when specifying a monitor address or port). Since monitors use monitor maps for discovery and they share monitor maps with clients and other Ceph daemons, the monitor map provides monitors with a strict guarantee that their consensus is valid.

Strict consistency also applies to updates to the monitor map. As with any other updates on the Ceph monitor, changes to the monitor map always run through a distributed consensus algorithm called Paxos. The Ceph monitors must agree on each update to the monitor map, such as adding or removing a Ceph monitor, to ensure that each monitor in the quorum has the same version of the monitor map. Updates to the monitor map are incremental so that Ceph monitors have the latest agreed upon version, and a set of previous versions. Maintaining a history enables a Ceph monitor that has an older version of the monitor map to catch up with the current state of the Red Hat Ceph Storage cluster.

If Ceph monitors discovered each other through the Ceph configuration file instead of through the monitor map, it would introduce additional risks because the Ceph configuration files are not updated and distributed automatically. Ceph monitors might inadvertently use an older Ceph configuration file, fail to recognize a Ceph monitor, fall out of a quorum, or develop a situation where Paxos is not able to determine the current state of the system accurately.

3.1.4. Bootstrapping Monitors

In most configuration and deployment cases, tools that deploy Ceph might help bootstrap the Ceph monitors by generating a monitor map for you, for example, Red Hat Storage Console or Ansible. A Ceph monitor requires a few explicit settings:

- **File System ID**: The **fsid** is the unique identifier for your object store. Since you can run multiple clusters on the same hardware, you must specify the unique ID of the object store when bootstrapping a monitor. Using deployment tools for example, Red Hat Storage Console or Ansible will generate a file system identifier, but you can specify the **fsid** manually too.

- **Monitor ID**: A monitor ID is a unique ID assigned to each monitor within the cluster. It is an alphanumeric value, and by convention the identifier usually follows an alphabetical increment (for example, **a**, **b**, and so on). This can be set in the Ceph configuration file (for example, **[mon.a]**, **[mon.b]**, and so on), by a deployment tool, or using the **ceph** command.

- **Keys**: The monitor must have secret keys.

3.2. CONFIGURING MONITORS

To apply configuration settings to the entire cluster, enter the configuration settings under the **[global]** section. To apply configuration settings to all monitors in the cluster, enter the configuration settings under the **[mon]** section. To apply configuration settings to specific monitors, specify the monitor instance (for example, **[mon.a]**). By convention, monitor instance names use alpha notation.

```
[global]

[mon]

[mon.a]

[mon.b]

[mon.c]
```

3.2.1. Minimum Configuration

The bare minimum monitor settings for a Ceph monitor in the Ceph configuration file includes a host name for each monitor if it is not configured for DNS and the monitor address. You can configure these under **[mon]** or under the entry for a specific monitor.

```
[mon]
mon_host = hostname1,hostname2,hostname3
mon_addr = 10.0.0.10:6789,10.0.0.11:6789,10.0.0.12:6789
```

Or

```
[mon.a]
host = hostname1
mon_addr = 10.0.0.10:6789
```

NOTE

This minimum configuration for monitors assumes that a deployment tool generates the **fsid** and the **mon.** key for you.

IMPORTANT

Once you deploy a Ceph cluster, do not change the IP address of the monitors.

As of RHCS 2.4, Ceph does not require the **mon_host** when the cluster is configured to look up a monitor via the DNS server. To configure the Ceph cluster for DNS lookup, set the **mon_dns_srv_name** setting in the Ceph configuration file.

mon_dns_srv_name

Description

The service name used for querying the DNS for the monitor hosts/addresses.

Type

String

Default

ceph-mon

Once set, configure the DNS. Create records either IPv4 (A) or IPv6 (AAAA) for the monitors in the DNS zone. For example:

```
#IPv4
mon1.example.com. A 192.168.0.1
mon2.example.com. A 192.168.0.2
mon3.example.com. A 192.168.0.3

#IPv6
mon1.example.com. AAAA 2001:db8::100
mon2.example.com. AAAA 2001:db8::200
mon3.example.com. AAAA 2001:db8::300
```

Where: **example.com** is the DNS search domain.

Then, create the SRV TCP records with the name **mon_dns_srv_name** configuration setting pointing to the three Monitors. The following example uses the default **ceph-mon** value.

```
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon1.example.com.
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon2.example.com.
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon3.example.com.
```

Monitors run on port **6789** by default, and their priority and weight are all set to **10** and **60** respectively in the foregoing example.

3.2.2. Cluster ID

Each Red Hat Ceph Storage cluster has a unique identifier (**fsid**). If specified, it usually appears under the **[global]** section of the configuration file. Deployment tools usually generate the **fsid** and store it in the monitor map, so the value may not appear in a configuration file. The **fsid** makes it possible to run daemons for multiple clusters on the same hardware.

fsid

Description

The cluster ID. One per cluster.

Type

UUID

Required

Yes.

Default

N/A. May be generated by a deployment tool if not specified.

NOTE

Do not set this value if you use a deployment tool that does it for you.

3.2.3. Initial Members

Red Hat recommends running a production Red Hat Ceph Storage cluster with at least three Ceph monitors to ensure high availability. When you run multiple monitors, you can specify the initial monitors that must be members of the cluster in order to establish a quorum. This may reduce the time it takes for the cluster to come online.

```
[mon]
mon_initial_members = a,b,c
```

mon_initial_members

Description

The IDs of initial monitors in a cluster during startup. If specified, Ceph requires an odd number of monitors to form an initial quorum (for example, 3).

Type

String

Default

None

NOTE

A *majority* of monitors in your cluster must be able to reach each other in order to establish a quorum. You can decrease the initial number of monitors to establish a quorum with this setting.

3.2.4. Data

Ceph provides a default path where Ceph monitors store data. For optimal performance in a production Red Hat Ceph Storage cluster, Red Hat recommends running Ceph monitors on separate hosts and drives from Ceph OSDs. Ceph monitors call the **fsync()** function often, which can interfere with Ceph

OSD workloads.

Ceph monitors store their data as key-value pairs. Using a data store prevents recovering Ceph monitors from running corrupted versions through Paxos, and it enables multiple modification operations in one single atomic batch, among other advantages.

NOTE

Red Hat does not recommend changing the default data location. If you modify the default location, make it uniform across Ceph monitors by setting it in the **[mon]** section of the configuration file.

mon_data

Description

The monitor's data location.

Type

String

Default

/var/lib/ceph/mon/$cluster-$id

mon_data_size_warn

Description

Ceph issues a **HEALTH_WARN** status in the cluster log when the monitor's data store reaches this threshold. The default value is 15GB.

Type

Integer

Default

15*1024*1024*1024*`

mon_data_avail_warn

Description

Ceph issues a **HEALTH_WARN** status in cluster log when the available disk space of the monitor's data store is lower than or equal to this percentage.

Type

Integer

Default

30

mon_data_avail_crit

Description

Ceph issues a **HEALTH_ERR** status in cluster log when the available disk space of the monitor's data store is lower or equal to this percentage.

Type

Integer

Default

5

mon_warn_on_cache_pools_without_hit_sets

Description

Ceph issues a **HEALTH_WARN** status in cluster log if a cache pool does not have the **hit_set_type** paramater set. See Pool Values for more details.

Type

Boolean

Default

True

mon_warn_on_crush_straw_calc_version_zero

Description

Ceph issues a **HEALTH_WARN** status in the cluster log if the CRUSH's **straw_calc_version** is zero. See CRUSH tunables for details.

Type

Boolean

Default

True

mon_warn_on_legacy_crush_tunables

Description

Ceph issues a **HEALTH_WARN** status in the cluster log if CRUSH tunables are too old (older than **mon_min_crush_required_version**).

Type

Boolean

Default

True

mon_crush_min_required_version

Description

This setting defines the minimum tunable profile version required by the cluster. See CRUSH tunables for details.

Type

String

Default

firefly

mon_warn_on_osd_down_out_interval_zero

Description

Ceph issues a **HEALTH_WARN** status in the cluster log if the **mon_osd_down_out_interval** setting is zero, because the Leader behaves in a similar manner when the **noout** flag is set. Administrators find it easier to troubleshoot a cluster by setting the **noout** flag. Ceph issues the warning to ensure administrators know that the setting is zero.

Type

Boolean

Default

True

mon_cache_target_full_warn_ratio

Description

Ceph issues a warning when between the ratio of **cache_target_full** and **target_max_object**.

Type

Float

Default

0.66

mon_health_data_update_interval

Description

How often (in seconds) a monitor in the quorum shares its health status with its peers. A negative number disables health updates.

Type

Float

Default

60

mon_health_to_clog

Description

This setting enable Ceph to send a health summary to the cluster log periodically.

Type

Boolean

Default

True

mon_health_to_clog_tick_interval

Description

How often (in seconds) the monitor sends a health summary to the cluster log. A non-positive number disables it. If the current health summary is empty or identical to the last time, the monitor will not send the status to the cluster log.

Type

Integer

Default

3600

mon_health_to_clog_interval

Description

How often (in seconds) the monitor sends a health summary to the cluster log. A non-positive number disables it. The monitor will always send the summary to cluster log.

Type

Integer

Default

```
60
```

3.2.5. Storage Capacity

When a Red Hat Ceph Storage cluster gets close to its maximum capacity (specifies by the **mon_osd_full_ratio** parameter), Ceph prevents you from writing to or reading from Ceph OSDs as a safety measure to prevent data loss. Therefore, letting a production Red Hat Ceph Storage cluster approach its full ratio is not a good practice, because it sacrifices high availability. The default full ratio is **.95**, or 95% of capacity. This a very aggressive setting for a test cluster with a small number of OSDs.

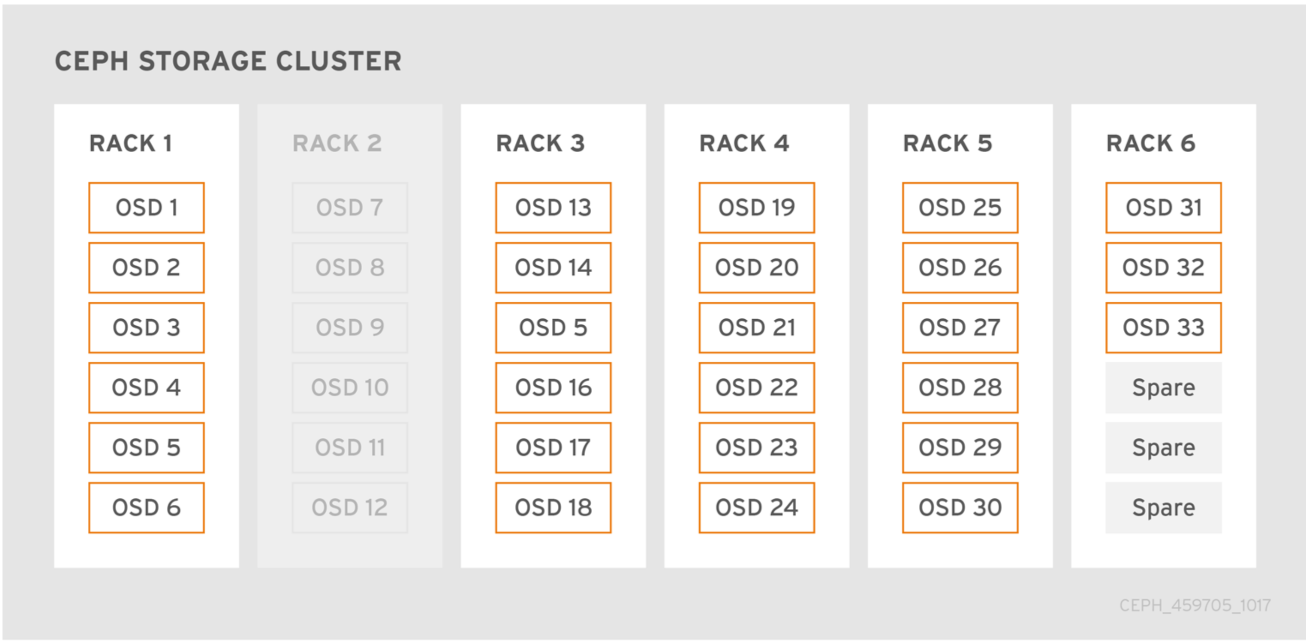
TIP

When monitoring a cluster, be alert to warnings related to the **nearfull** ratio. This means that a failure of some OSDs could result in a temporary service disruption if one or more OSDs fails. Consider adding more OSDs to increase storage capacity.

A common scenario for test clusters involves a system administrator removing a Ceph OSD from the Red Hat Ceph Storage cluster to watch the cluster re-balance. Then, removing another Ceph OSD, and so on until the Red Hat Ceph Storage cluster eventually reaches the full ratio and locks up.

Red Hat recommends a bit of capacity planning even with a test cluster. Planning enables you to gauge how much spare capacity you will need in order to maintain high availability. Ideally, you want to plan for a series of Ceph OSD failures where the cluster can recover to an **active + clean** state without replacing those Ceph OSDs immediately. You can run a cluster in an **active + degraded** state, but this is not ideal for normal operating conditions.

The following diagram depicts a simplistic Red Hat Ceph Storage cluster containing 33 Ceph Nodes with one Ceph OSD per host, each Ceph OSD Daemon reading from and writing to a 3TB drive. So this exemplary Red Hat Ceph Storage cluster has a maximum actual capacity of 99TB. With a **mon osd full ratio** of **0.95**, if the Red Hat Ceph Storage cluster falls to 5 TB of remaining capacity, the cluster will not allow Ceph clients to read and write data. So the Red Hat Ceph Storage cluster's operating capacity is 95 TB, not 99 TB.



It is normal in such a cluster for one or two OSDs to fail. A less frequent but reasonable scenario involves a rack's router or power supply failing, which brings down multiple OSDs simultaneously (for example, OSDs 7-12). In such a scenario, you should still strive for a cluster that can remain operational and achieve an **active + clean** state, even if that means adding a few hosts with additional OSDs in short order. If your capacity utilization is too high, you might not lose data, but you could still sacrifice data availability while resolving an outage within a failure domain if capacity utilization of the cluster exceeds the full ratio. For this reason, Red Hat recommends at least some rough capacity planning.

Identify two numbers for your cluster:

- the number of OSDs
- the total capacity of the cluster

To determine the mean average capacity of an OSD within a cluster, divide the total capacity of the cluster by the number of OSDs in the cluster. Consider multiplying that number by the number of OSDs you expect to fail simultaneously during normal operations (a relatively small number). Finally, multiply the capacity of the cluster by the full ratio to arrive at a maximum operating capacity. Then, subtract the number of amount of data from the OSDs you expect to fail to arrive at a reasonable full ratio. Repeat the foregoing process with a higher number of OSD failures (for example, a rack of OSDs) to arrive at a reasonable number for a near full ratio.

```
[global]
...
mon_osd_full_ratio = .80
mon_osd_nearfull_ratio = .70
```

mon_osd_full_ratio

Description

The percentage of disk space used before an OSD is considered **full**.

Type

Float:

Default

.95

mon_osd_nearfull_ratio

Description

The percentage of disk space used before an OSD is considered **nearfull**.

Type

Float

Default

.85

TIP

If some OSDs are **nearfull**, but others have plenty of capacity, you might have a problem with the CRUSH weight for the **nearfull** OSDs.

3.2.6. Heartbeat

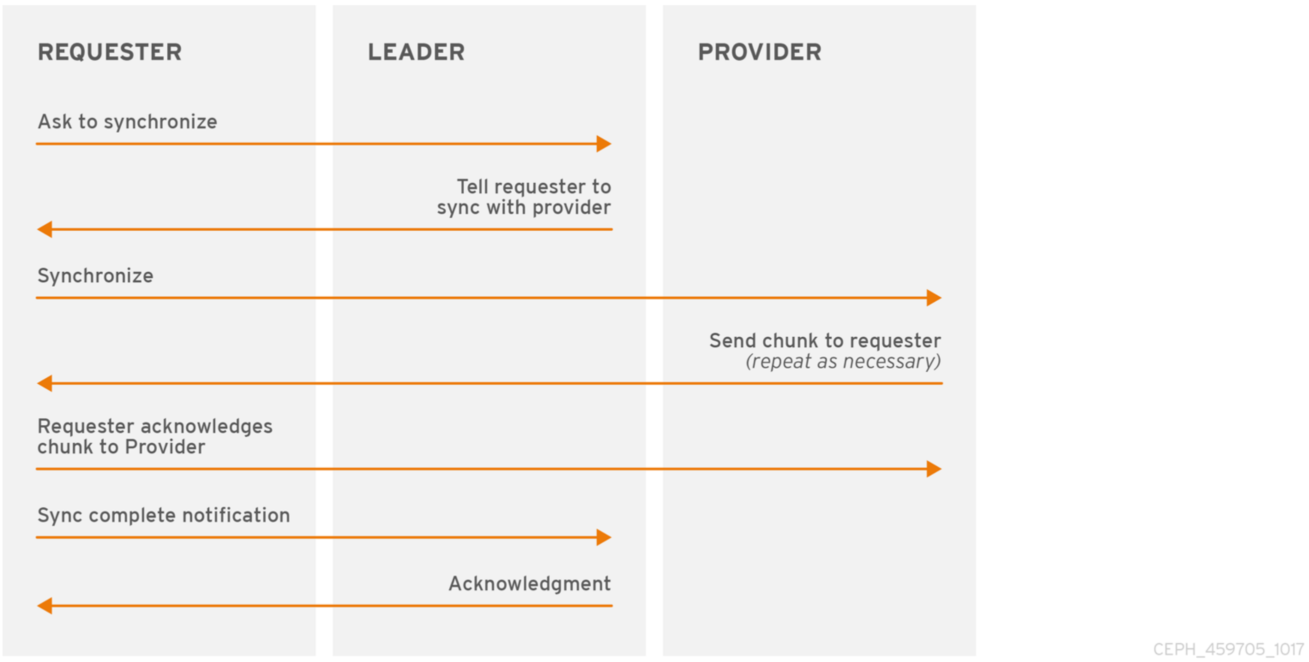
Ceph monitors know about the cluster by requiring reports from each OSD, and by receiving reports from OSDs about the status of their neighboring OSDs. Ceph provides reasonable default settings for interaction between monitor and OSD, however, you can modify them as needed.

3.2.7. Monitor Store Synchronization

When you run a production cluster with multiple monitors which is recommended, each monitor checks to see if a neighboring monitor has a more recent version of the cluster map. For example, a map in a neighboring monitor with one or more epoch numbers higher than the most current epoch in the map of the instant monitor. Periodically, one monitor in the cluster might fall behind the other monitors to the point where it must leave the quorum, synchronize to retrieve the most current information about the cluster, and then rejoin the quorum. For the purposes of synchronization, monitors can assume one of three roles:

- **Leader**: The Leader is the first monitor to achieve the most recent Paxos version of the cluster map.
- **Provider**: The Provider is a monitor that has the most recent version of the cluster map, but was not the first to achieve the most recent version.
- **Requester:** The Requester is a monitor that has fallen behind the leader and must synchronize in order to retrieve the most recent information about the cluster before it can rejoin the quorum.

These roles enable a leader to delegate synchronization duties to a provider, which prevents synchronization requests from overloading the leader and improving performance. In the following diagram, the requester has learned that it has fallen behind the other monitors. The requester asks the leader to synchronize, and the leader tells the requester to synchronize with a provider.


REQUESTER
LEADER
PROVIDER
Ask to synchronize
Tell requester to sync with provider
Synchronize
Send chunk to requester (repeat as necessary)
Requester acknowledges chunk to Provider
Sync complete notification
Acknowledgment
CEPH_459705_1017

Synchronization always occurs when a new monitor joins the cluster. During runtime operations, monitors can receive updates to the cluster map at different times. This means the leader and provider roles may migrate from one monitor to another. If this happens while synchronizing (for example, a provider falls behind the leader), the provider can terminate synchronization with a requester.

Once synchronization is complete, Ceph requires trimming across the cluster. Trimming requires that the placement groups are **active + clean**.

mon_sync_trim_timeout

Description, Type

Double

Default

30.0

mon_sync_heartbeat_timeout

Description, Type

Double

Default

30.0

mon_sync_heartbeat_interval

Description, Type

Double

Default

5.0

mon_sync_backoff_timeout

Description, Type

Double

Default

30.0

mon_sync_timeout

Description

Number of seconds the monitor will wait for the next update message from its sync provider before it gives up and bootstraps again.

Type

Double

Default

30.0

mon_sync_max_retries

Description, Type

Integer

Default

5

mon_sync_max_payload_size

Description

The maximum size for a sync payload (in bytes).

Type

32-bit Integer

Default

1045676

paxos_max_join_drift

Description

The maximum Paxos iterations before we must first sync the monitor data stores. When a monitor finds that its peer is too far ahead of it, it will first sync with data stores before moving on.

Type

Integer

Default

10

paxos_stash_full_interval

Description

How often (in commits) to stash a full copy of the PaxosService state. Current this setting only affects **mds**, **mon**, **auth** and **mgr** PaxosServices.

Type

Integer

Default

25

paxos_propose_interval

Description

Gather updates for this time interval before proposing a map update.

Type

Double

Default

1.0

paxos_min

Description

The minimum number of paxos states to keep around

Type

Integer

Default

500

paxos_min_wait

Description

The minimum amount of time to gather updates after a period of inactivity.

Type

Double

Default

0.05

paxos_trim_min

Description

Number of extra proposals tolerated before trimming

Type

Integer

Default

250

paxos_trim_max

Description

The maximum number of extra proposals to trim at a time

Type

Integer

Default

500

paxos_service_trim_min

Description

The minimum amount of versions to trigger a trim (0 disables it)

Type

Integer

Default

250

paxos_service_trim_max

Description

The maximum amount of versions to trim during a single proposal (0 disables it)

Type

Integer

Default

500

mon_max_log_epochs

Description

The maximum amount of log epochs to trim during a single proposal

Type

Integer

Default

500

mon_max_pgmap_epochs

Description

The maximum amount of pgmap epochs to trim during a single proposal

Type
: Integer

Default
: 500

mon_mds_force_trim_to

Description
: Force monitor to trim mdsmaps to this point (0 disables it. dangerous, use with care)

Type
: Integer

Default
: 0

mon_osd_force_trim_to

Description
: Force monitor to trim osdmaps to this point, even if there is PGs not clean at the specified epoch (0 disables it. dangerous, use with care)

Type
: Integer

Default
: 0

mon_osd_cache_size

Description
: The size of osdmaps cache, not to rely on underlying store's cache

Type
: Integer

Default
: 10

mon_election_timeout

Description
: On election proposer, maximum waiting time for all ACKs in seconds.

Type
: Float

Default
: **5**

mon_lease

Description
: The length (in seconds) of the lease on the monitor's versions.

Type
: Float

Default

5

mon_lease_renew_interval_factor

Description

mon lease * **mon lease renew interval factor** will be the interval for the Leader to renew the other monitor's leases. The factor should be less than **1.0**.

Type

Float

Default

0.6

mon_lease_ack_timeout_factor

Description

The Leader will wait **mon lease** * **mon lease ack timeout factor** for the Providers to acknowledge the lease extension.

Type

Float

Default

2.0

mon_accept_timeout_factor

Description

The Leader will wait **mon lease** * **mon accept timeout factor** for the Requester(s) to accept a Paxos update. It is also used during the Paxos recovery phase for similar purposes.

Type

Float

Default

2.0

mon_min_osdmap_epochs

Description

Minimum number of OSD map epochs to keep at all times.

Type

32-bit Integer

Default

500

mon_max_pgmap_epochs

Description

Maximum number of PG map epochs the monitor should keep.

Type

32-bit Integer

Default

500

mon_max_log_epochs

Description

Maximum number of Log epochs the monitor should keep.

Type

32-bit Integer

Default

500

3.2.8. Clock

Ceph daemons pass critical messages to each other, which must be processed before daemons reach a timeout threshold. If the clocks in Ceph monitors are not synchronized, it can lead to a number of anomalies. For example:

- Daemons ignoring received messages (for example, timestamps outdated).
- Timeouts triggered too soon or late when a message was not received in time.

See Monitor Store Synchronization for details.

TIP

Install NTP on the Ceph monitor hosts to ensure that the monitor cluster operates with synchronized clocks.

Clock drift may still be noticeable with NTP even though the discrepancy is not yet harmful. Ceph clock drift and clock skew warnings can get triggered even though NTP maintains a reasonable level of synchronization. Increasing your clock drift may be tolerable under such circumstances. However, a number of factors such as workload, network latency, configuring overrides to default timeouts and the Monitor Store Synchronization settings may influence the level of acceptable clock drift without compromising Paxos guarantees.

Ceph provides the following tunable options to allow you to find acceptable values.

clock_offset

Description

How much to offset the system clock. See **Clock.cc** for details.

Type

Double

Default

0

mon_tick_interval

Description

A monitor’s tick interval in seconds.

Type

32-bit Integer

Default

5

mon_clock_drift_allowed

Description

The clock drift in seconds allowed between monitors.

Type

Float

Default

.050

mon_clock_drift_warn_backoff

Description

Exponential backoff for clock drift warnings.

Type

Float

Default

5

mon_timecheck_interval

Description

The time check interval (clock drift check) in seconds for the leader.

Type

Float

Default

300.0

mon_timecheck_skew_interval

Description

The time check interval (clock drift check) in seconds when in the presence of a skew in seconds for the Leader.

Type

Float

Default

30.0

3.2.9. Client

mon_client_hunt_interval

Description

The client will try a new monitor every **N** seconds until it establishes a connection.

Type

Double

Default

3.0

mon_client_ping_interval

Description

The client will ping the monitor every **N** seconds.

Type

Double

Default

10.0

mon_client_max_log_entries_per_message

Description

The maximum number of log entries a monitor will generate per client message.

Type

Integer

Default

1000

mon_client_bytes

Description

The amount of client message data allowed in memory (in bytes).

Type

64-bit Integer Unsigned

Default

100ul << 20

3.3. MISCELLANEOUS

mon_max_osd

Description

The maximum number of OSDs allowed in the cluster.

Type

32-bit Integer

Default

10000

mon_globalid_prealloc

Description

The number of global IDs to pre-allocate for clients and daemons in the cluster.

Type

32-bit Integer

Default

100

mon_sync_fs_threshold

Description

Synchronize with the filesystem when writing the specified number of objects. Set it to **0** to disable it.

Type

32-bit Integer

Default

5

mon_subscribe_interval

Description

The refresh interval (in seconds) for subscriptions. The subscription mechanism enables obtaining the cluster maps and log information.

Type

Double

Default

300

mon_stat_smooth_intervals

Description

Ceph will smooth statistics over the last **N** PG maps.

Type

Integer

Default

2

mon_probe_timeout

Description

Number of seconds the monitor will wait to find peers before bootstrapping.

Type

Double

Default

2.0

mon_daemon_bytes

Description

The message memory cap for metadata server and OSD messages (in bytes).

Type

64-bit Integer Unsigned

Default

400ul << 20

mon_max_log_entries_per_event

Description

The maximum number of log entries per event.

Type

Integer

Default

4096

mon_osd_prime_pg_temp

Description

Enables or disable priming the PGMap with the previous OSDs when an out OSD comes back into the cluster. With the **true** setting the clients will continue to use the previous OSDs until the newly in OSDs as that PG peered.

Type

Boolean

Default

true

mon_osd_prime_pg_temp_max_time

Description

How much time in seconds the monitor should spend trying to prime the PGMap when an out OSD comes back into the cluster.

Type

Float

Default

0.5

mon_osd_prime_pg_temp_max_time_estimate

Description

Maximum estimate of time spent on each PG before we prime all PGs in parallel.

Type

Float

Default

0.25

mon_osd_allow_primary_affinity

Description

allow **primary_affinity** to be set in the osdmap.

Type

Boolean

Default

False

mon_osd_pool_ec_fast_read

Description

Whether turn on fast read on the pool or not. It will be used as the default setting of newly created erasure pools if **fast_read** is not specified at create time.

Type

Boolean

Default

False

mon_mds_skip_sanity

Description

Skip safety assertions on FSMap (in case of bugs where we want to continue anyway). Monitor terminates if the FSMap sanity check fails, but we can disable it by enabling this option.

Type

Boolean

Default

False

mon_max_mdsmap_epochs

Description

The maximum amount of mdsmap epochs to trim during a single proposal.

Type

Integer

Default

500

mon_config_key_max_entry_size

Description

The maximum size of config-key entry (in bytes)

Type

Integer

Default

4096

mon_scrub_interval

Description

How often (in seconds) the monitor scrub its store by comparing the stored checksums with the computed ones of all the stored keys.

Type

Integer

Default

3600*24

mon_scrub_max_keys

Description

The maximum number of keys to scrub each time.

Type

Integer

Default

100

mon_compact_on_start

Description

Compact the database used as Ceph Monitor store on **ceph-mon** start. A manual compaction helps to shrink the monitor database and improve the performance of it if the regular compaction fails to work.

Type

Boolean

Default

False

mon_compact_on_bootstrap

Description

Compact the database used as Ceph Monitor store on on bootstrap. Monitor starts probing each other for creating a quorum after bootstrap. If it times out before joining the quorum, it will start over and bootstrap itself again.

Type

Boolean

Default

False

mon_compact_on_trim

Description

Compact a certain prefix (including paxos) when we trim its old states.

Type

Boolean

Default

True

mon_cpu_threads

Description

Number of threads for performing CPU intensive work on monitor.

Type

Boolean

Default

True

mon_osd_mapping_pgs_per_chunk

Description

We calculate the mapping from placement group to OSDs in chunks. This option specifies the number of placement groups per chunk.

Type

Integer

Default

4096

mon_osd_max_split_count

Description

Largest number of PGs per "involved" OSD to let split create. When we increase the **pg_num** of a pool, the placement groups will be splitted on all OSDs serving that pool. We want to avoid extreme multipliers on PG splits.

Type

Integer

Default

300

mon_session_timeout

Description

Monitor will terminate inactive sessions stay idle over this time limit.

Type

Integer

Default

300

rados_mon_op_timeout

Description

Number of seconds that RADOS waits for a response from the Ceph Monitor before returning an error from a RADOS operation. A value of 0 means no limit.

Type

Double

Default

0

CHAPTER 4. CEPHX CONFIGURATION REFERENCE

The **cephx** protocol is enabled by default. Cryptographic authentication has some computational costs, though they are generally quite low. If the network environment connecting a client and server hosts is very safe and you cannot afford authentication, you can disable it. However, Red Hat recommends using authentication.

NOTE

If you disable authentication, you are at risk of a man-in-the-middle attack altering client and server messages, which could lead to significant security issues.

4.1. MANUAL

When you deploy a cluster manually, you have to bootstrap the monitor manually and create the **client.admin** user and keyring. To deploy Ceph manually, see our Knowledgebase article. The steps for monitor bootstrapping are the logical steps you must perform when using third party deployment tools like Chef, Puppet, Juju, and so on.

4.2. ENABLING AND DISABLING CEPHX

Enabling Cephx requires that you have deployed keys for your monitors and OSDs. If you are simply toggling Cephx on / off, you do not have to repeat the bootstrapping procedures.

4.2.1. Enabling Cephx

When **cephx** is enabled, Ceph will look for the keyring in the default search path, which includes **/etc/ceph/$cluster.$name.keyring**. You can override this location by adding a **keyring** option in the **[global]** section of the Ceph configuration file, but this is not recommended.

Execute the following procedures to enable **cephx** on a cluster with authentication disabled. If you or your deployment utility have already generated the keys, you may skip the steps related to generating keys.

1. Create a **client.admin** key, and save a copy of the key for your client host:

```
ceph auth get-or-create client.admin mon 'allow *' osd 'allow *' -o
/etc/ceph/ceph.client.admin.keyring
```

WARNING

This will erase the contents of any existing **/etc/ceph/client.admin.keyring** file. Do not perform this step if a deployment tool has already done it for you.

2. Create a keyring for the monitor cluster and generate a monitor secret key:

```
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'
```

3. Copy the monitor keyring into a **ceph.mon.keyring** file in every monitor **mon data** directory. For example, to copy it to **mon.a** in cluster **ceph**, use the following:

   ```
   cp /tmp/ceph.mon.keyring /var/lib/ceph/mon/ceph-a/keyring
   ```

4. Generate a secret key for every OSD, where **{$id}** is the OSD number:

   ```
   ceph auth get-or-create osd.{$id} mon 'allow rwx' osd 'allow *' -o /var/lib/ceph/osd/ceph-
   {$id}/keyring
   ```

5. By default the **cephx** authentication protocol is enabled.

   NOTE

   If the **cephx** authentication protocol was disabled previously by setting the authentication options to **none**, then by removing the following lines under the **[global]** section in the Ceph configuration file (**/etc/ceph/ceph.conf**) will reenable the **cephx** authentication protocol:

   ```
   auth_cluster_required = none
   auth_service_required = none
   auth_client_required = none
   ```

6. Start or restart the Ceph cluster.

   IMPORTANT

   Enabling **cephx** requires downtime because the cluster needs to be completely restarted, or it needs to be shut down and then started while client I/O is disabled.

   These flags need to be set before restarting or shutting down the storage cluster:

   ```
   # ceph osd set noout
   # ceph osd set norecover
   # ceph osd set norebalance
   # ceph osd set nobackfill
   # ceph osd set nodown
   # ceph osd set pause
   ```

   Once **cephx** is enabled and all PGs are active and clean, unset the flags:

   ```
   # ceph osd unset noout
   # ceph osd unset norecover
   # ceph osd unset norebalance
   # ceph osd unset nobackfill
   # ceph osd unset nodown
   # ceph osd unset pause
   ```

4.2.2. Disabling Cephx

The following procedure describes how to disable Cephx. If your cluster environment is relatively safe, you can offset the computation expense of running authentication. Red Hat recommends enabling authentication. However, it may be easier during setup or troubleshooting to temporarily disable authentication.

1. Disable **cephx** authentication by setting the following options in the **[global]** section of the Ceph configuration file:

   ```
   auth_cluster_required = none
   auth_service_required = none
   auth_client_required = none
   ```

2. Start or restart the Ceph cluster.

4.3. CONFIGURATION SETTINGS

4.3.1. Enablement

auth_cluster_required

Description

If enabled, the Red Hat Ceph Storage cluster daemons (that is, **ceph-mon** and **ceph-osd**) must authenticate with each other. Valid settings are **cephx** or **none**.

Type

String

Required

No

Default

cephx.

auth_service_required

Description

If enabled, the Red Hat Ceph Storage cluster daemons require Ceph clients to authenticate with the Red Hat Ceph Storage cluster in order to access Ceph services. Valid settings are **cephx** or **none**.

Type

String

Required

No

Default

cephx.

auth_client_required

Description

If enabled, the Ceph client requires the Red Hat Ceph Storage cluster to authenticate with the Ceph client. Valid settings are **cephx** or **none**.

Type

String

Required

No

Default

cephx.

4.3.2. Keys

When you run Ceph with authentication enabled, the **ceph** administrative commands and Ceph clients require authentication keys to access the Ceph storage cluster.

The most common way to provide these keys to the **ceph** administrative commands and clients is to include a Ceph keyring under the **/etc/ceph/** directory. The file name is usually **ceph.client.admin.keyring** or **$cluster.client.admin.keyring**. If you include the keyring under the **/etc/ceph/** directory, you do not need to specify a **keyring** entry in the Ceph configuration file.

Red Hat recommends copying the Red Hat Ceph Storage cluster keyring file to nodes where you will run administrative commands, because it contains the **client.admin** key. To do so, execute the following command as **root**:

```
# scp <user>@<hostname>:/etc/ceph/ceph.client.admin.keyring /etc/ceph/ceph.client.admin.keyring
```

Replace **<user>** with the user name used on the host with the **client.admin** key and **<hostname>** with the host name of that host.

NOTE

Ensure the **ceph.keyring** file has appropriate permissions set on the client machine.

You can specify the key itself in the Ceph configuration file using the **key** setting, which is not recommended, or a path to a key file using the **keyfile** setting.

keyring

Description

The path to the keyring file.

Type

String

Required

No

Default

/etc/ceph/$cluster.$name.keyring,/etc/ceph/$cluster.keyring,/etc/ceph/keyring,/etc/ceph/keyring.bin

keyfile

Description

The path to a key file (that is. a file containing only the key).

Type

String

Required

No

Default

None

key

Description

The key (that is, the text string of the key itself). Not recommended.

Type

String

Required

No

Default

None

4.3.3. Daemon Keyrings

Administrative users or deployment tools might generate daemon keyrings in the same way as generating user keyrings. By default, Ceph stores daemons keyrings inside their data directory. The default keyring locations, and the capabilities necessary for the daemon to function, are shown below.

ceph-mon

Location

$mon_data/keyring

Capabilities

mon 'allow *'

ceph-osd

Location

$osd_data/keyring

Capabilities

mon 'allow profile osd' osd 'allow *'

radosgw

Location

$rgw_data/keyring

Capabilities

mon 'allow rwx' osd 'allow rwx'

NOTE

The monitor keyring (that is **mon.**) contains a key but no capabilities, and is not part of the cluster **auth** database.

The daemon data directory locations default to directories of the form:

```
/var/lib/ceph/$type/$cluster-$id
```

-

For example, **osd.12** is:

```
/var/lib/ceph/osd/ceph-12
```

You can override these locations, but it is not recommended.

4.3.4. Signatures

Red Hat recommends that Ceph authenticate all ongoing messages between the entities using the session key set up for that initial authentication.

Like other parts of Ceph authentication, Ceph provides fine-grained control so you can enable or disable signatures for service messages between the client and Ceph, and you can enable or disable signatures for messages between Ceph daemons.

cephx_require_signatures

Description

If set to **true**, Ceph requires signatures on all message traffic between the Ceph client and the Red Hat Ceph Storage cluster, and between daemons comprising the Red Hat Ceph Storage cluster.

Type

Boolean

Required

No

Default

false

cephx_cluster_require_signatures

Description

If set to **true**, Ceph requires signatures on all message traffic between Ceph daemons comprising the Red Hat Ceph Storage cluster.

Type

Boolean

Required

No

Default

false

cephx_service_require_signatures

Description

If set to **true**, Ceph requires signatures on all message traffic between Ceph clients and the Red Hat Ceph Storage cluster.

Type

Boolean

Required

No

Default

false

cephx_sign_messages

Description

If the Ceph version supports message signing, Ceph will sign all messages so they cannot be spoofed.

Type

Boolean

Default

true

NOTE

Ceph kernel modules do not support signatures yet.

4.3.5. Time to Live

auth_service_ticket_ttl

Description

When the Red Hat Ceph Storage cluster sends a Ceph client a ticket for authentication, the cluster assigns the ticket a time to live.

Type

Double

Default

60*60

CHAPTER 5. POOL, PG, AND CRUSH CONFIGURATION REFERENCE

When you create pools and set the number of placement groups for the pool, Ceph uses default values when you do not specifically override the defaults. Red Hat recommends overriding some of the defaults. Specifically, set a pool's replica size and override the default number of placement groups. You can set these values when running pool commands. You can also override the defaults by adding new ones in the **[global]** section of the Ceph configuration file.

```
[global]

# By default, Ceph makes 3 replicas of objects. If you want to set 4
# copies of an object as the default value--a primary copy and three replica
# copies--reset the default values as shown in 'osd pool default size'.
# If you want to allow Ceph to write a lesser number of copies in a degraded
# state, set 'osd pool default min size' to a number less than the
# 'osd pool default size' value.

osd_pool_default_size = 4  # Write an object 4 times.
osd_pool_default_min_size = 1 # Allow writing one copy in a degraded state.

# Ensure you have a realistic number of placement groups. We recommend
# approximately 100 per OSD. E.g., total number of OSDs multiplied by 100
# divided by the number of replicas (i.e., osd pool default size). So for
# 10 OSDs and osd pool default size = 4, we'd recommend approximately
# (100 * 10) / 4 = 250.

osd_pool_default_pg_num = 250
osd_pool_default_pgp_num = 250
```

5.1. SETTINGS

mon_allow_pool_delete

Description

Allows a monitor to delete a pool. In RHCS 3 and later releases, the monitor cannot delete the pool by default as an added measure to protect data.

Type

Boolean

Default

false

mon_max_pool_pg_num

Description

The maximum number of placement groups per pool.

Type

Integer

Default

65536

mon_pg_create_interval

Description
: Number of seconds between PG creation in the same Ceph OSD Daemon.

Type
: Float

Default
: **30.0**

mon_pg_stuck_threshold

Description
: Number of seconds after which PGs can be considered as being stuck.

Type
: 32-bit Integer

Default
: **300**

mon_pg_min_inactive

Description
: Ceph issues a **HEALTH_ERR** status in the cluster log if the number of PGs that remain inactive longer than the **mon_pg_stuck_threshold** exceeds this setting. The default setting is one PG. A non-positive number disables this setting.

Type
: Integer

Default
: **1**

mon_pg_warn_min_per_osd

Description
: Ceph issues a **HEALTH_WARN** status in the cluster log if the average number of PGs per OSD in the cluster is less than this setting. A non-positive number disables this setting.

Type
: Integer

Default
: **30**

mon_pg_warn_max_per_osd

Description
: Ceph issues a **HEALTH_WARN** status in the cluster log if the average number of PGs per OSD in the cluster is greater than this setting. A non-positive number disables this setting.

Type
: Integer

Default
: **300**

mon_pg_warn_min_objects

Description

Do not warn if the total number of objects in the cluster is below this number.

Type

Integer

Default

1000

mon_pg_warn_min_pool_objects

Description

Do not warn on pools whose object number is below this number.

Type

Integer

Default

1000

mon_pg_check_down_all_threshold

Description

The threshold of **down** OSDs by percentage after which Ceph checks all PGs to ensure they are not stuck or stale.

Type

Float

Default

0.5

mon_pg_warn_max_object_skew

Description

Ceph issue a **HEALTH_WARN** status in the cluster log if the average number of objects in a pool is greater than **mon pg warn max object skew** times the average number of objects for all pools. A non-positive number disables this setting.

Type

Float

Default

10

mon_delta_reset_interval

Description

The number of seconds of inactivity before Ceph resets the PG delta to zero. Ceph keeps track of the delta of the used space for each pool to aid administrators in evaluating the progress of recovery and performance.

Type

Integer

Default

10

mon_osd_max_op_age

Description

The maximimum age in seconds for an operation to complete before issuing a **HEALTH_WARN** status.

Type

Float

Default

32.0

osd_pg_bits

Description

Placement group bits per Ceph OSD Daemon.

Type

32-bit Integer

Default

6

osd_pgp_bits

Description

The number of bits per Ceph OSD Daemon for Placement Groups for Placement purpose (PGPs).

Type

32-bit Integer

Default

6

osd_crush_chooseleaf_type

Description

The bucket type to use for **chooseleaf** in a CRUSH rule. Uses ordinal rank rather than name.

Type

32-bit Integer

Default

1. Typically a host containing one or more Ceph OSD Daemons.

osd_pool_default_crush_replicated_ruleset

Description

The default CRUSH ruleset to use when creating a replicated pool.

Type

8-bit Integer

Default

0

osd_pool_erasure_code_stripe_unit

Description

Sets the default size, in bytes, of a chunk of an object stripe for erasure coded pools. Every object of size S will be stored as N stripes, with each data chunk receiving **stripe unit** bytes. Each stripe of **N * stripe unit** bytes will be encoded/decoded individually. This option can is overridden by the **stripe_unit** setting in an erasure code profile.

Type

Unsigned 32-bit Integer

Default

4096

osd_pool_default_size

Description

Sets the number of replicas for objects in the pool. The default value is the same as **ceph osd pool set {pool-name} size {size}**.

Type

32-bit Integer

Default

3

osd_pool_default_min_size

Description

Sets the minimum number of written replicas for objects in the pool in order to acknowledge a write operation to the client. If minimum is not met, Ceph will not acknowledge the write to the client. This setting ensures a minimum number of replicas when operating in **degraded** mode.

Type

32-bit Integer

Default

0, which means no particular minimum. If **0**, minimum is **size - (size / 2)**.

osd_pool_default_pg_num

Description

The default number of placement groups for a pool. The default value is the same as **pg_num** with **mkpool**.

Type

32-bit Integer

Default

8

osd_pool_default_pgp_num

Description

The default number of placement groups for placement for a pool. The default value is the same as **pgp_num** with **mkpool**. PG and PGP should be equal (for now).

Type

32-bit Integer

Default

8

osd_pool_default_flags

Description

The default flags for new pools.

Type

32-bit Integer

Default

0

osd_max_pgls

Description

The maximum number of placement groups to list. A client requesting a large number can tie up the Ceph OSD Daemon.

Type

Unsigned 64-bit Integer

Default

1024

Note

Default should be fine.

osd_min_pg_log_entries

Description

The minimum number of placement group logs to maintain when trimming log files.

Type

32-bit Int Unsigned

Default

1000

osd_default_data_pool_replay_window

Description

The time (in seconds) for an OSD to wait for a client to replay a request.

Type

32-bit Integer

Default

45

CHAPTER 6. OSD CONFIGURATION REFERENCE

You can configure Ceph OSDs in the Ceph configuration file, but Ceph OSDs can use the default values and a very minimal configuration. A minimal Ceph OSD configuration sets the **osd journal size** and **osd host** options, and uses default values for almost everything else.

Ceph OSDs are numerically identified in incremental fashion, beginning with **0** using the following convention:

```
osd.0
osd.1
osd.2
```

In a configuration file, you can specify settings for all Ceph OSDs in the cluster by adding configuration settings to the **[osd]** section of the configuration file. To add settings directly to a particular Ceph OSD (for example, **osd host**), enter it in a section specific only to that OSD in the Ceph configuration file. For example:

```
[osd]
osd journal size = 1024

[osd.0]
osd host = osd-host-a

[osd.1]
osd host = osd-host-b
```

6.1. GENERAL SETTINGS

The following settings provide a Ceph OSD's ID, and determine paths to data and journals. Ceph deployment scripts typically generate the UUID automatically.

IMPORTANT

Red Hat does not recommend changing the default paths for data or journals, as it makes it more problematic to troubleshoot Ceph later.

The journal size should be at least twice the product of the expected drive speed multiplied by the value of the **filestore max sync interval** option. However, the most common practice is to partition the journal drive (often an SSD), and mount it such that Ceph uses the entire partition for the journal.

osd_uuid

Description

The universally unique identifier (UUID) for the Ceph OSD.

Type

UUID

Default

The UUID.

Note

The **osd uuid** applies to a single Ceph OSD. The **fsid** applies to the entire cluster.

osd_data

Description

The path to the OSD’s data. You must create the directory when deploying Ceph. Mount a drive for OSD data at this mount point. Red Hat does not recommend changing the default.

Type

String

Default

/var/lib/ceph/osd/$cluster-$id

osd_max_write_size

Description

The maximum size of a write in megabytes.

Type

32-bit Integer

Default

90

osd_client_message_size_cap

Description

The largest client data message allowed in memory.

Type

64-bit Integer Unsigned

Default

500MB default. **500*1024L*1024L**

osd_class_dir

Description

The class path for RADOS class plug-ins.

Type

String

Default

$libdir/rados-classes

6.2. JOURNAL SETTINGS

By default, Ceph expects that you will store a Ceph OSD’s journal with the following path:

```
/var/lib/ceph/osd/$cluster-$id/journal
```

Without performance optimization, Ceph stores the journal on the same disk as the Ceph OSD’s data. A Ceph OSD optimized for performance can use a separate disk to store journal data, for example, a solid state drive delivers high performance journaling.

A journal size should find the product of the **filestore max sync interval** and the expected throughput, and multiply the product by two (2):

```
osd journal size = <2 * (expected throughput * filestore max sync interval)>
```

The expected throughput number should include the expected disk throughput (that is, sustained data transfer rate), and network throughput. For example, a 7200 RPM disk will likely have approximately 100 MB/s. Taking the **min()** of the disk and network throughput should provide a reasonable expected throughput. Some users just start off with a 10GB journal size. For example:

```
osd journal size = 10000
```

> **WARNING**
>
> Sizing the journal correctly for your OSDs is important. Using a small journal will lead to a slower recovery in the event of an OSD failure. The number of recovery threads has to be decreased in order to have a stable recovery by keeping pressure in journal at an acceptable level. Also, committing transactions to the file store will be slower and could lead to the file store hanging if the queued transaction size is bigger than the journal size.

osd_journal

Description

The path to the OSD's journal. This may be a path to a file or a block device (such as a partition of an SSD). If it is a file, you must create the directory to contain it. We recommend using a drive separate from the **osd data** drive.

Type

String

Default

/var/lib/ceph/osd/$cluster-$id/journal

osd_journal_size

Description

The size of the journal in megabytes. If this is 0, and the journal is a block device, the entire block device is used. This is ignored if the journal is a block device, and the entire block device is used.

Type

32-bit Integer

Default

5120

Recommended

Begin with 1GB. Should be at least twice the product of the expected speed multiplied by **filestore max sync interval**.

6.3. SCRUBBING

In addition to making multiple copies of objects, Ceph insures data integrity by scrubbing placement groups. Ceph scrubbing is analogous to the **fsck** command on the object storage layer.

For each placement group, Ceph generates a catalog of all objects and compares each primary object and its replicas to ensure that no objects are missing or mismatched.

Light scrubbing (daily) checks the object size and attributes. Deep scrubbing (weekly) reads the data and uses checksums to ensure data integrity.

Scrubbing is important for maintaining data integrity, but it can reduce performance. Adjust the following settings to increase or decrease scrubbing operations.

osd_max_scrubs

Description

The maximum number of simultaneous scrub operations for a Ceph OSD.

Type

32-bit Int

Default

1

osd_scrub_thread_timeout

Description

The maximum time in seconds before timing out a scrub thread.

Type

32-bit Integer

Default

60

osd_scrub_finalize_thread_timeout

Description

The maximum time in seconds before timing out a scrub finalize thread.

Type

32-bit Integer

Default

60*10

osd_scrub_begin_hour

Description

The earliest hour that light or deep scrubbing can begin. It is used with the **osd scrub end hour** parameter to define a scrubbing time window and allows constraining scrubbing to off-peak hours. The setting takes an integer to specify the hour on the 24-hour cycle where **0** represents the hour from 12:01 a.m. to 1:00 a.m., 13 represents the hour from 1:01 p.m. to 2:00 p.m., and so on.

Type

32-bit Integer

Default

0 for 12:01 to 1:00 a.m.

osd_scrub_end_hour

Description

The latest hour that light or deep scrubbing can begin. It is used with the **osd scrub begin hour** parameter to define a scrubbing time window and allows constraining scrubbing to off-peak hours. The setting takes an integer to specify the hour on the 24-hour cycle where **0** represents the hour from 12:01 a.m. to 1:00 a.m., 13 represents the hour from 1:01 p.m. to 2:00 p.m., and so on. The **end** hour must be greater than the **begin** hour.

Type

32-bit Integer

Default

24 for 11:01 p.m. to 12:00 a.m.

osd_scrub_load_threshold

Description

The maximum load. Ceph will not scrub when the system load (as defined by the **getloadavg()** function) is higher than this number. Default is **0.5**.

Type

Float

Default

0.5

osd_scrub_min_interval

Description

The minimum interval in seconds for scrubbing the Ceph OSD when the Red Hat Ceph Storage cluster load is low.

Type

Float

Default

Once per day. **60*60*24**

osd_scrub_max_interval

Description

The maximum interval in seconds for scrubbing the Ceph OSD irrespective of cluster load.

Type

Float

Default

Once per week. **7*60*60*24**

osd_scrub_interval_randomize_ratio

Description

Takes the ratio and randomizes the scheduled scrub between **osd scrub min interval** and **osd scrub max interval**.

Type

Float

Default

0.5.

mon_warn_not_scrubbed

Description

Number of seconds after **osd_scrub_interval** to warn about any PGs that were not scrubbed.

Type

Integer

Default

0 (no warning).

osd_scrub_chunk_min

Description

The object store is partitioned into chunks which end on hash boundaries. For chunky scrubs, Ceph scrubs objects one chunk at a time with writes blocked for that chunk. The **osd scrub chunk min** setting represents minimum number of chunks to scrub.

Type

32-bit Integer

Default

5

osd_scrub_chunk_max

Description

The maximum number of chunks to scrub.

Type

32-bit Integer

Default

25

osd_scrub_sleep

Description

The time to sleep between deep scrub operations.

Type

Float

Default

0 (or off).

osd_scrub_during_recovery

Description

Allows scrubbing during recovery.

Type

Bool

Default

false

osd_scrub_invalid_stats

Description

Forces extra scrub to fix stats marked as invalid.

Type

Bool

Default

true

osd_scrub_priority

Description

Controls queue priority of scrub operations versus client I/O.

Type

Unsigned 32-bit Integer

Default

5

osd_scrub_cost

Description

Cost of scrub operations in megabytes for queue scheduling purposes.

Type

Unsigned 32-bit Integer

Default

50 << 20

osd_deep_scrub_interval

Description

The interval for deep scrubbing, that is fully reading all data. The **osd scrub load threshold** parameter does not affect this setting.

Type

Float

Default

Once per week. **60*60*24*7**

osd_deep_scrub_stride

Description

Read size when doing a deep scrub.

Type

32-bit Integer

Default

512 KB. **524288**

mon_warn_not_deep_scrubbed

Description

Number of seconds after **osd_deep_scrub_interval** to warn about any PGs that were not scrubbed.

Type

Integer

Default

0 (no warning).

osd_deep_scrub_randomize_ratio

Description

The rate at which scrubs will randomly become deep scrubs (even before **osd_deep_scrub_interval** has past).

Type

Float

Default

0.15 or 15%.

osd_deep_scrub_update_digest_min_age

Description

How many seconds old objects must be before scrub updates the whole-object digest.

Type

Integer

Default

120 (2 hours).

6.4. OPERATIONS

Operations settings allow you to configure the number of threads for servicing requests.

By default, Ceph uses two threads with a 30 second timeout and a 30 second complaint time if an operation does not complete within those time parameters. Set operations priority weights between client operations and recovery operations to ensure optimal performance during recovery.

osd_op_num_shards

Description

The number of shards for client operations.

Type

32-bit Integer

Default

0

osd_op_num_threads_per_shard

Description

The number of threads per shard for client operations.

Type

32-bit Integer

Default

0

osd_op_num_shards_hdd

Description

The number of shards for HDD operations.

Type

32-bit Integer

Default

5

osd_op_num_threads_per_shard_hdd

Description

The number of threads per shard for HDD operations.

Type

32-bit Integer

Default

1

osd_op_num_shards_ssd

Description

The number of shards for SSD operations.

Type

32-bit Integer

Default

8

osd_op_num_threads_per_shard_ssd

Description

The number of threads per shard for SSD operations.

Type

32-bit Integer

Default

2

osd_client_op_priority

Description

The priority set for client operations. It is relative to **osd recovery op priority**.

Type

32-bit Integer

Default

63

Valid Range

1-63

osd_recovery_op_priority

Description

The priority set for recovery operations. It is relative to **osd client op priority**.

Type

32-bit Integer

Default

3

Valid Range

1-63

osd_op_thread_timeout

Description

The Ceph OSD operation thread timeout in seconds.

Type

32-bit Integer

Default

30

osd_op_complaint_time

Description

An operation becomes complaint worthy after the specified number of seconds have elapsed.

Type

Float

Default

30

osd_disk_threads

Description

The number of disk threads, which are used to perform background disk intensive OSD operations such as scrubbing and snap trimming.

Type

32-bit Integer

Default

1

osd_disk_thread_ioprio_class

Description

Sets the **ioprio_set(2)** I/O scheduling **class** for the disk thread. Acceptable values are:

- **idle**
- **be**

- **rt**

  The **idle** class means the disk thread will have lower priority than any other thread in the OSD. This is useful to slow down scrubbing on an OSD that is busy handling client operations.

  The **be** class is the default and is the same priority as all other threads in the OSD.

  The **rt** class means the disk thread will have precedence over all other threads in the OSD. This is useful if scrubbing is much needed and must make progress at the expense of client operations.

Type

String

Default

an empty string

osd_disk_thread_ioprio_priority

Description

It sets the **ioprio_set(2)** I/O scheduling **priority** of the disk thread ranging from 0 (highest) to 7 (lowest). If all OSDs on a given host were in class **idle** and compete for I/O due to controller congestion, it can be used to lower the disk thread priority of one OSD to 7 so that another OSD with priority 0 can potentially scrub faster.

Type

Integer in the range of 0 to 7 or -1 if not to be used.

Default

-1

IMPORTANT

The **osd disk thread ioprio class** and **osd disk thread ioprio priority** options will only be used if both are set to a non default value. In addition, it only works with the Linux Kernel CFQ scheduler.

osd_op_history_size

Description

The maximum number of completed operations to track.

Type

32-bit Unsigned Integer

Default

20

osd_op_history_duration

Description

The oldest completed operation to track.

Type

32-bit Unsigned Integer

Default

600

osd_op_log_threshold

Description

How many operations logs to display at once.

Type

32-bit Integer

Default

5

osd_op_timeout

Description

The time in seconds after which running OSD operations time out.

Type

Integer

Default

0

IMPORTANT

Do not set the **osd op timeout** option unless your clients can handle the consequences. For example, setting this parameter on clients running in virtual machines can lead to data corruption because the virtual machines interpret this timeout as a hardware failure.

6.5. BACKFILLING

When you add Ceph OSDs to a cluster or remove them from the cluster, the CRUSH algorithm rebalances the cluster by moving placement groups to or from Ceph OSDs to restore the balance. The process of migrating placement groups and the objects they contain can reduce the cluster operational performance considerably. To maintain operational performance, Ceph performs this migration with the 'backfill' process, which allows Ceph to set backfill operations to a lower priority than requests to read or write data.

osd_max_backfills

Description

The maximum number of backfill operations allowed to or from a single OSD.

Type

64-bit Unsigned Integer

Default

1

osd_backfill_scan_min

Description

The minimum number of objects per backfill scan.

Type

32-bit Integer

Default

64

osd_backfill_scan_max

Description

The maximum number of objects per backfill scan.

Type

32-bit Integer

Default

512

osd_backfillfull_ratio

Description

Refuse to accept backfill requests when the Ceph OSD's full ratio is above this value.

Type

Float

Default

0.85

osd_backfill_retry_interval

Description

The number of seconds to wait before retrying backfill requests.

Type

Double

Default

10.0

6.6. OSD MAP

OSD maps reflect the OSD daemons operating in the cluster. Over time, the number of map epochs increases. Ceph provides the following settings to ensure that Ceph performs well as the OSD map grows larger.

osd_map_dedup

Description

Enable removing duplicates in the OSD map.

Type

Boolean

Default

true

osd_map_cache_size

Description

The size of the OSD map cache in megabytes.

Type

32-bit Integer

Default

50

osd_map_cache_bl_size

Description

The size of the in-memory OSD map cache in OSD daemons.

Type

32-bit Integer

Default

50

osd_map_cache_bl_inc_size

Description

The size of the in-memory OSD map cache incrementals in OSD daemons.

Type

32-bit Integer

Default

100

osd_map_message_max

Description

The maximum map entries allowed per MOSDMap message.

Type

32-bit Integer

Default

40

6.7. RECOVERY

When the cluster starts or when a Ceph OSD terminates unexpectedly and restarts, the OSD begins peering with other Ceph OSDs before write operation can occur.

If a Ceph OSD crashes and comes back online, usually it will be out of sync with other Ceph OSDs containing more recent versions of objects in the placement groups. When this happens, the Ceph OSD goes into recovery mode and seeks to get the latest copy of the data and bring its map back up to date. Depending upon how long the Ceph OSD was down, the OSD's objects and placement groups may be significantly out of date. Also, if a failure domain went down (for example, a rack), more than one Ceph OSD may come back online at the same time. This can make the recovery process time consuming and resource intensive.

To maintain operational performance, Ceph performs recovery with limitations on the number recovery requests, threads and object chunk sizes which allows Ceph perform well in a degraded state.

osd_recovery_delay_start

Description

After peering completes, Ceph will delay for the specified number of seconds before starting to recover objects.

Type

Float

Default

0

osd_recovery_max_active

Description

The number of active recovery requests per OSD at one time. More requests will accelerate recovery, but the requests places an increased load on the cluster.

Type

32-bit Integer

Default

3

osd_recovery_max_chunk

Description

The maximum size of a recovered chunk of data to push.

Type

64-bit Integer Unsigned

Default

8 << 20

osd_recovery_threads

Description

The number of threads for recovering data.

Type

32-bit Integer

Default

1

osd_recovery_thread_timeout

Description

The maximum time in seconds before timing out a recovery thread.

Type

32-bit Integer

Default

30

osd_recover_clone_overlap

Description

Preserves clone overlap during recovery. Should always be set to **true**.

Type

Boolean

Default

true

6.8. MISCELLANEOUS

osd_snap_trim_thread_timeout

Description

The maximum time in seconds before timing out a snap trim thread.

Type

32-bit Integer

Default

60*60*1

osd_pg_max_concurrent_snap_trims

Description

The max number of parallel snap trims/PG. This controls how many objects per PG to trim at once.

Type

32-bit Integer

Default

2

osd_snap_trim_sleep

Description

Insert a sleep between every trim operation a PG issues.

Type

32-bit Integer

Default

0

osd_max_trimming_pgs

Description

The max number of trimming PGs

Type

32-bit Integer

Default

2

osd_backlog_thread_timeout

Description

The maximum time in seconds before timing out a backlog thread.

Type

32-bit Integer

Default

60*60*1

osd_default_notify_timeout

Description

The OSD default notification timeout (in seconds).

Type

32-bit Integer Unsigned

Default

30

osd_check_for_log_corruption

Description

Check log files for corruption. Can be computationally expensive.

Type

Boolean

Default

false

osd_remove_thread_timeout

Description

The maximum time in seconds before timing out a remove OSD thread.

Type

32-bit Integer

Default

60*60

osd_command_thread_timeout

Description

The maximum time in seconds before timing out a command thread.

Type

32-bit Integer

Default

10*60

osd_command_max_records

Description

Limits the number of lost objects to return.

Type

32-bit Integer

Default

256

osd_auto_upgrade_tmap

Description

Uses **tmap** for **omap** on old objects.

Type

Boolean

Default

true

osd_tmapput_sets_users_tmap

Description

Uses **tmap** for debugging only.

Type

Boolean

Default

false

osd_preserve_trimmed_log

Description

Preserves trimmed log files, but uses more disk space.

Type

Boolean

Default

false

rados_osd_op_timeout

Description

Number of seconds that RADOS waits for a response from the OSD before returning an error from a RADOS operation. A value of 0 means no limit.

Type

Double

Default

0

CHAPTER 7. CONFIGURING MONITOR AND OSD INTERACTION

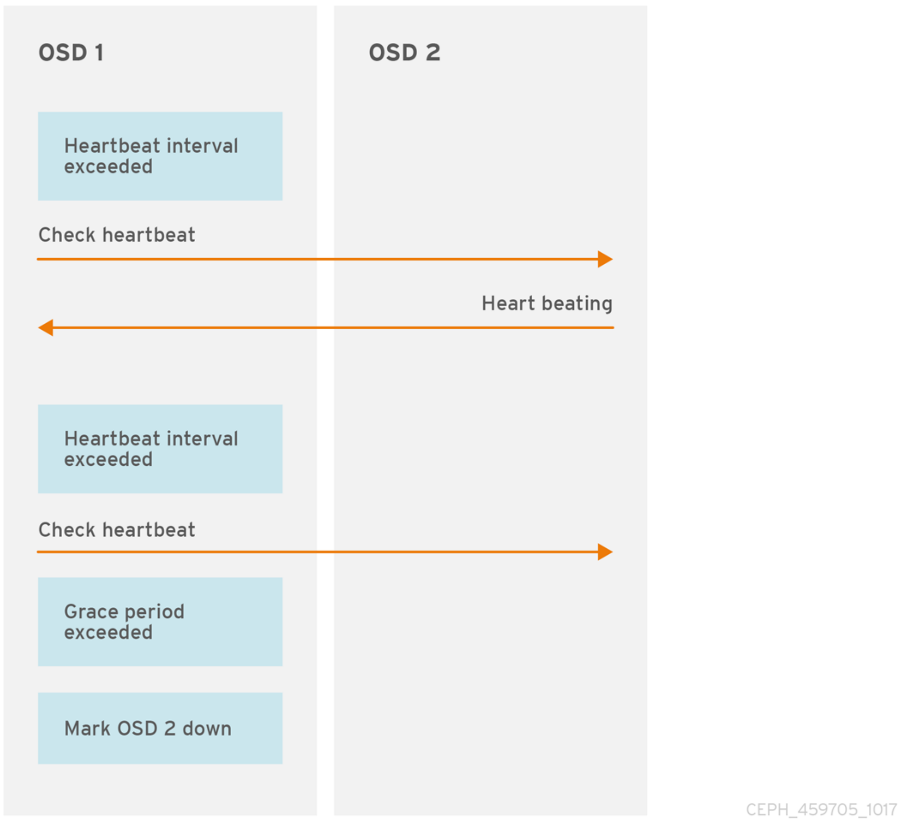
After you have completed your initial Ceph configuration, you can deploy and run Ceph. When you execute a command such as **ceph health** or **ceph -s**, the Ceph Monitor reports on the current state of the Ceph Storage Cluster. The Ceph Monitor knows about the Ceph Storage Cluster by requiring reports from each Ceph OSD Daemon, and by receiving reports from Ceph OSD Daemons about the status of their neighboring Ceph OSD Daemons. If the Ceph Monitor does not receive reports, or if it receives reports of changes in the Ceph Storage Cluster, the Ceph Monitor updates the status of the Ceph Cluster Map.

Ceph provides reasonable default settings for Ceph Monitor and Ceph OSD Daemon interaction. However, you can override the defaults. The following sections describe how Ceph Monitors and Ceph OSD Daemons interact for the purposes of monitoring the Ceph Storage Cluster.

7.1. OSDS CHECK HEARTBEATS

Each Ceph OSD Daemon checks the heartbeat of other Ceph OSD Daemons every 6 seconds. To change the heartbeat interval, add the **osd heartbeat interval** setting under the **[osd]** section of the Ceph configuration file, or change its value at runtime.

If a neighboring Ceph OSD Daemon does not send heartbeat packets within a 20 second grace period, the Ceph OSD Daemon might consider the neighboring Ceph OSD Daemon **down** and report it back to a Ceph Monitor, which will update the Ceph Cluster Map. To change this grace period, add the **osd heartbeat grace** setting under the **[osd]** section of the Ceph configuration file, or set its value at runtime.



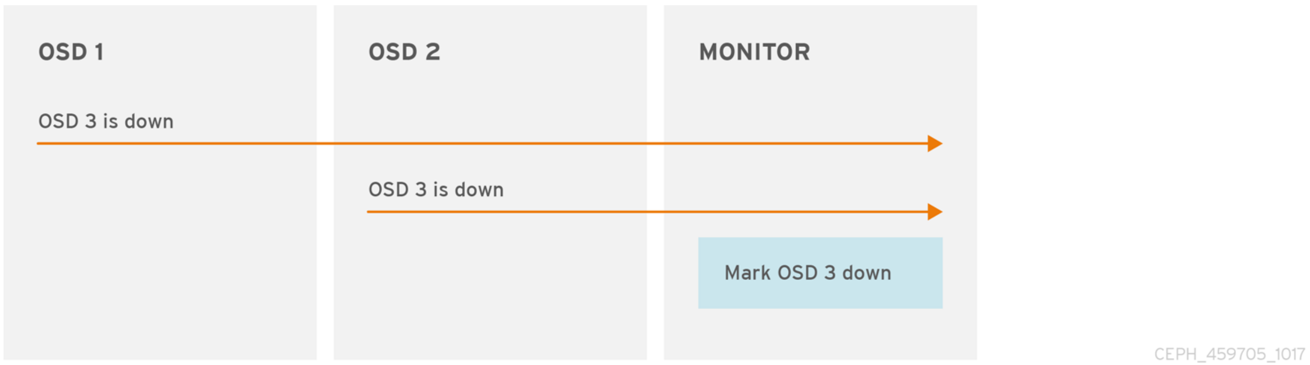
7.2. OSDS REPORT DOWN OSDS

By default, two Ceph OSD Daemons **from different hosts** must report to the Ceph Monitors that another Ceph OSD Daemon is **down** before the Ceph Monitors acknowledge that the reported Ceph OSD Daemon is **down**.

However, there is chance that all the OSDs reporting the failure are in different hosts in a rack with a bad switch that causes connection problems between OSDs.

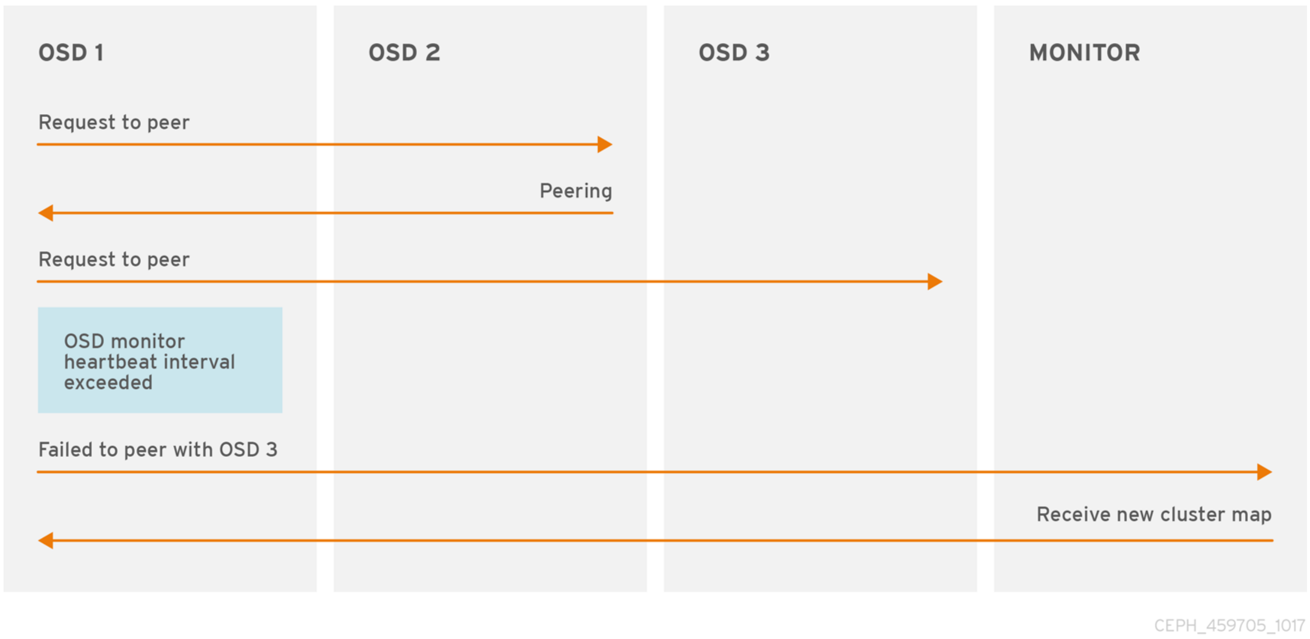
To avoid a "false alarm," Ceph considers the peers reporting the failure as a proxy for a "subcluster" that is similarly laggy. While this is not always the case, it may help administrators localize the grace correction to a subset of the system that is performing poorly.

Ceph uses the **mon_osd_reporter_subtree_level** setting to group the peers into the "subcluster" by their common ancestor type in the CRUSH map. By default, only two reports from **a different subtree** are required to report another Ceph OSD Daemon **down**. Administrators can change the number of reporters from unique subtrees and the common ancestor type required to report a Ceph OSD Daemon **down** to a Ceph Monitor by adding the **mon_osd_min_down_reporters** and **mon_osd_reporter_subtree_level** settings under the **[mon]** section of the Ceph configuration file, or by setting the value at runtime.



7.3. OSDS REPORT PEERING FAILURE

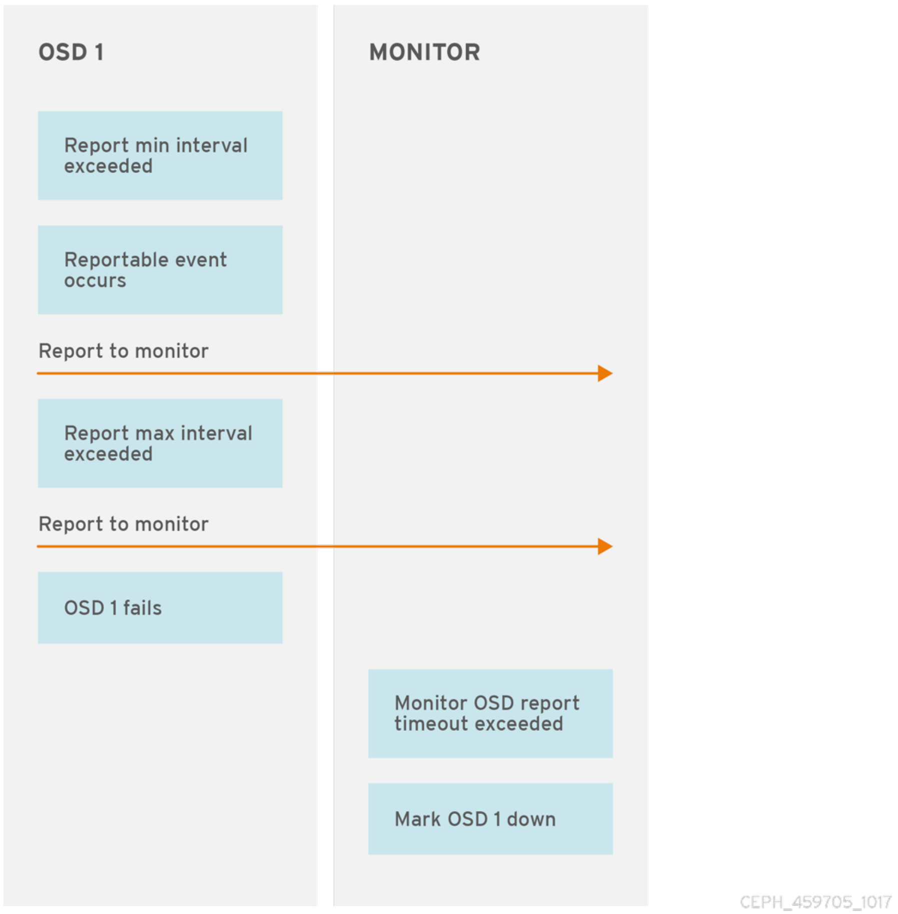
If a Ceph OSD Daemon cannot peer with any of the Ceph OSD Daemons defined in its Ceph configuration file or the cluster map, it will ping a Ceph Monitor for the most recent copy of the cluster map every 30 seconds. You can change the Ceph Monitor heartbeat interval by adding the **osd mon heartbeat interval** setting under the **[osd]** section of the Ceph configuration file, or by setting the value at runtime.



7.4. OSDS REPORT THEIR STATUS

If an Ceph OSD Daemon does not report to a Ceph Monitor, the Ceph Monitor will consider the Ceph OSD Daemon **down** after the **mon osd report timeout** elapses. A Ceph OSD Daemon sends a report to a Ceph Monitor when a reportable event such as a failure, a change in placement group stats, a change in **up_thru** or when it boots within 5 seconds. You can change the Ceph OSD Daemon minimum report interval by adding the **osd mon report interval min** setting under the **[osd]** section of the Ceph configuration file, or by setting the value at runtime.

A Ceph OSD Daemon sends a report to a Ceph Monitor every 120 seconds irrespective of whether any notable changes occur. You can change the Ceph Monitor report interval by adding the **osd mon report interval max** setting under the **[osd]** section of the Ceph configuration file, or by setting the value at runtime.



7.5. CONFIGURATION SETTINGS

When modifying heartbeat settings, include them in the **[global]** section of the Ceph configuration file.

7.5.1. Monitor Settings

mon_osd_min_up_ratio

Description

The minimum ratio of **up** Ceph OSD Daemons before Ceph will mark Ceph OSD Daemons **down**.

Type

Double

Default

.3

mon_osd_min_in_ratio

Description

The minimum ratio of **in** Ceph OSD Daemons before Ceph will mark Ceph OSD Daemons **out**.

Type

Double

Default

.3

mon_osd_laggy_halflife

Description

The number of seconds **laggy** estimates will decay.

Type

Integer

Default

60*60

mon_osd_laggy_weight

Description

The weight for new samples in **laggy** estimation decay.

Type

Double

Default

0.3

mon_osd_laggy_max_interval

Description

Maximum value of **laggy_interval** in laggy estimations (in seconds). The monitor uses an adaptive approach to evaluate the **laggy_interval** of a certain OSD. This value will be used to calculate the grace time for that OSD.

Type

Integer

Default

300

mon_osd_adjust_heartbeat_grace

Description

If set to **true**, Ceph will scale based on **laggy** estimations.

Type

Boolean

Default

true

mon_osd_adjust_down_out_interval

Description
: If set to **true**, Ceph will scaled based on **laggy** estimations.

Type
: Boolean

Default
: **true**

mon_osd_auto_mark_in

Description
: Ceph will mark any booting Ceph OSD Daemons as **in** the Ceph Storage Cluster.

Type
: Boolean

Default
: **false**

mon_osd_auto_mark_auto_out_in

Description
: Ceph will mark booting Ceph OSD Daemons auto marked **out** of the Ceph Storage Cluster as **in** the cluster.

Type
: Boolean

Default
: **true**

mon_osd_auto_mark_new_in

Description
: Ceph will mark booting new Ceph OSD Daemons as **in** the Ceph Storage Cluster.

Type
: Boolean

Default
: **true**

mon_osd_down_out_interval

Description
: The number of seconds Ceph waits before marking a Ceph OSD Daemon **down** and **out** if it does not respond.

Type
: 32-bit Integer

Default
: **600**

mon_osd_downout_subtree_limit

Description

The largest CRUSH unit type that Ceph will automatically mark **out**.

Type

String

Default

rack

mon_osd_reporter_subtree_level

Description

This setting defines the parent CRUSH unit type for the reporting OSDs. The OSDs send failure reports to the monitor if they find an unresponsive peer. The monitor may mark the reported OSD **down** and then **out** after a grace period.

Type

String

Default

host

mon_osd_report_timeout

Description

The grace period in seconds before declaring unresponsive Ceph OSD Daemons **down**.

Type

32-bit Integer

Default

900

mon_osd_min_down_reporters

Description

The minimum number of Ceph OSD Daemons required to report a **down** Ceph OSD Daemon.

Type

32-bit Integer

Default

2

7.5.2. OSD Settings

osd_heartbeat_address

Description

An Ceph OSD Daemon's network address for heartbeats.

Type

Address

Default

The host address.

osd_heartbeat_interval

Description

How often an Ceph OSD Daemon pings its peers (in seconds).

Type

32-bit Integer

Default

6

osd_heartbeat_grace

Description

The elapsed time when a Ceph OSD Daemon has not shown a heartbeat that the Ceph Storage Cluster considers it **down**.

Type

32-bit Integer

Default

20

osd_mon_heartbeat_interval

Description

How often the Ceph OSD Daemon pings a Ceph Monitor if it has no Ceph OSD Daemon peers.

Type

32-bit Integer

Default

30

osd_mon_report_interval_max

Description

The maximum time in seconds that a Ceph OSD Daemon can wait before it must report to a Ceph Monitor.

Type

32-bit Integer

Default

120

osd_mon_report_interval_min

Description

The minimum number of seconds a Ceph OSD Daemon may wait from startup or another reportable event before reporting to a Ceph Monitor.

Type

32-bit Integer

Default

5

Valid Range

Should be less than **osd mon report interval max**

osd_mon_ack_timeout

Description

The number of seconds to wait for a Ceph Monitor to acknowledge a request for statistics.

Type

32-bit Integer

Default

30

CHAPTER 8. FILE STORE CONFIGURATION REFERENCE

8.1. EXTENDED ATTRIBUTES

Extended attributes (XATTRs) are an important aspect in the CephFS configuration. Some file systems have limits on the number of bytes stored in extended attributes. Additionally, in some cases, the file system might not be as fast as an alternative method of storing extended attributes. The following settings improve CephFS performance by using a method of storing extended attributes that is extrinsic to the underlying file system.

Ceph extended attributes are stored as **inline xattr**, using the extended attributes provided by the underlying file system, if it does not impose a size limit. If there is a size limit (4KB total on ext4, for instance), some Ceph extended attributes will be stored in an key-value database called **omap** when the **filestore max inline xattr size** or **filestore max inline xattrs** threshold are reached.

filestore_xattr_use_omap

Description

Use object map for XATTRS. Set to **true** for ext4 file systems.

Type

Boolean

Required

No

Default

false

filestore_omap_header_cache_size

Description

Determines the size of the LRU used to cache object **omap** headers. Larger values use more memory but can reduce lookups on **omap**. (Experts only).

Type

Integer

Default

1024

filestore_omap_backend

Description

Used to determine which back end is used for the **omap**. Can be set to **leveldb** or **rocksdb**. (Experts only. **rocksdb** is experimental.)

Type

String

Default

leveldb

filestore_debug_omap_check

Description

Debugging check on synchronization. Expensive. For debugging only.

Type

Boolean

Required

No

Default

0

filestore_max_inline_xattr_size

Description

The maximum size of an extended attribute stored in a file system (that is, XFS, btrfs, ext4, and others) per object. Should not be larger than the file system can handle.

Type

Unsigned 32-bit Integer

Required

No

Default

512

filestore_max_inline_xattrs

Description

The maximum number of extended attributes stored in the file system per object.

Type

32-bit Integer

Required

No

Default

2

filestore_max_inline_xattr_size_xfs

Description

The maximum size of an extended attribute stored in the file system for XFS file systems per object. Should not be larger than the file system can handle.

Type

Unsigned 32-bit Integer

Default

65536

filestore_max_inline_xattr_size_btrfs

Description

The maximum size of an extended attribute stored in the file system for btrfs per object. Should not be larger than the file system can handle.

Type

Unsigned 32-bit Integer

Default

2048

filestore_max_inline_xattr_size_other

Description

The maximum size of an extended attribute stored in the file system for file systems other than btrfs or XFS per object. Should not be larger than the file system can handle.

Type

Unsigned 32-bit Integer

Default

512

filestore_max_inline_xattrs

Description

The maximum number of extended attributes stored in the file system per object. Overrides fine-grained settings.

Type

Unsigned 32-bit Integer

Default

0

filestore_max_inline_xattrs_xfs

Description

The maximum number of extended attributes stored in an XFS file system per object.

Type

Unsigned 32-bit Integer

Default

10

filestore_max_inline_xattrs_btrfs

Description

The maximum number of extended attributes stored in a btrfs file system per object.

Type

Unsigned 32-bit Integer

Default

10

filestore_max_inline_xattrs_other

Description

The maximum number of extended attributes stored in file systems other than btrfs or XFS per object.

Type

Unsigned 32-bit Integer

Default

2

8.2. SYNCHRONIZATION INTERVALS

Periodically, the file store needs to quiesce write operations and synchronize the file system, which creates a consistent commit point. It can then free journal entries up to the commit point. Synchronizing more frequently tends to reduce the time required to perform synchronization, and reduces the amount of data that needs to remain in the journal. Less frequent synchronization allows the backing file system to coalesce small writes and metadata updates more optimally—potentially resulting in more efficient synchronization.

filestore_max_sync_interval

Description

The maximum interval in seconds for synchronizing the file store.

Type

Double

Required

No

Default

5

filestore_min_sync_interval

Description

The minimum interval in seconds for synchronizing the file store.

Type

Double

Required

No

Default

.01

8.3. FLUSHER

The file store flusher forces data from large write operations to be written out using the **sync file range** option before the synchronization in order to reduce the cost of the eventual synchronization. In practice, disabling the file store flusher seems to improve performance in some cases.

filestore_flusher

Description

Enables the file store flusher.

Type

Boolean

Required

No

Default

false

filestore_flusher_max_fds

Description

Sets the maximum number of file descriptors for the flusher.

Type

Integer

Required

No

Default

512

filestore_sync_flush

Description

Enables the synchronization flusher.

Type

Boolean

Required

No

Default

false

filestore_fsync_flushes_journal_data

Description

Flush journal data during file system synchronization.

Type

Boolean

Required

No

Default

false

8.4. QUEUE

The following settings provide limits on the size of the file store queue.

filestore_queue_max_ops

Description

Defines the maximum number of operations in progress that the file store accepts before blocking on queuing new operations.

Type

Integer

Required

No. Minimal impact on performance.

Default

500

filestore_queue_max_bytes

Description

The maximum number of bytes for an operation.

Type

Integer

Required

No

Default

100 << 20

filestore_queue_committing_max_ops

Description

The maximum number of operations that the file store can commit.

Type

Integer

Required

No

Default

500

filestore_queue_committing_max_bytes

Description

The maximum number of bytes that the file store can commit.

Type

Integer

Required

No

Default

100 << 20

8.5. WRITEBACK THROTTLE

Ceph replicates some of the write-back behavior in the kernel, because the page cache tends to keep dirty data round too long.

filestore_wbthrottle_enable

Description

Enables the file store write-back throttle. The file store write-back throttle is used to prevent large amounts of uncommitted data from building up before each file store sync. (Experts only).

Type

Boolean

Default

true

filestore_wbthrottle_btrfs_bytes_start_flusher

Description

Dirty bytes threshold at which Ceph begins background flushing for the btrfs file system.

Type

64-bit Unsigned Integer

Default

41943040

filestore_wbthrottle_btrfs_bytes_hard_limit

Description

Dirty bytes threshold at which Ceph begins to throttle I/O until the flusher catches up for btrfs.

Type

64-bit Unsigned Integer

Default

419430400

filestore_wbthrottle_btrfs_ios_start_flusher

Description

Dirty I/Os threshold at which Ceph begins background flushing for btrfs.

Type

64-bit Unsigned Integer

Default

500

filestore_wbthrottle_btrfs_ios_hard_limit

Description

Dirty I/Os threshold at which Ceph begins to throttle IO until the flusher catches up for btrfs.

Type

64-bit Unsigned Integer

Default

5000

filestore_wbthrottle_btrfs_inodes_start_flusher

Description

Dirty inodes threshold at which Ceph begins background flushing for btrfs.

Type

64-bit Unsigned Integer

Default

500

filestore_wbthrottle_btrfs_inodes_hard_limit

Description

Dirty inodes threshold at which Ceph begins to throttle IO until the flusher catches up for btrfs. Must be less than the **fd** limit.

Type

64-bit Unsigned Integer

Default

5000

filestore_wbthrottle_xfs_bytes_start_flusher

Description

Dirty bytes threshold at which Ceph begins background flushing for the XFS file system.

Type

64-bit Unsigned Integer

Default

41943040

filestore_wbthrottle_xfs_bytes_hard_limit

Description

Dirty bytes threshold at which Ceph begins to throttle IO until the flusher catches up for XFS.

Type

64-bit Unsigned Integer

Default

419430400

filestore_wbthrottle_xfs_ios_start_flusher

Description

Dirty I/Os threshold at which Ceph begins background flushing for XFS.

Type

64-bit Unsigned Integer

Default

500

filestore_wbthrottle_xfs_ios_hard_limit

Description

Dirty I/Os threshold at which Ceph begins to throttle IO until the flusher catches up for XFS.

Type

64-bit Unsigned Integer

Default

5000

filestore_wbthrottle_xfs_inodes_start_flusher

Description

Dirty inodes threshold at which Ceph begins background flushing for XFS.

Type

64-bit Unsigned Integer

Default

500

filestore_wbthrottle_xfs_inodes_hard_limit

Description

Dirty inodes threshold at which Ceph begins to throttle IO until the flusher catches up for XFS. Must be less than the **fd** limit.

Type

64-bit Unsigned Integer

Default

5000

8.6. TIMEOUTS

filestore_op_threads

Description

The number of file system operation threads that execute in parallel.

Type

Integer

Required

No

Default

2

filestore_op_thread_timeout

Description

The timeout for a file system operation thread (in seconds).

Type

Integer

Required

No

Default

60

filestore_op_thread_suicide_timeout

Description

The timeout for a commit operation before canceling the commit (in seconds).

Type

Integer

Required

No

Default

180

8.7. B-TREE FILE SYSTEM

filestore_btrfs_snap

Description

Enable snapshots for a btrfs file store.

Type

Boolean

Required

No. Only used for btrfs.

Default

true

filestore_btrfs_clone_range

Description

Enable cloning ranges for a btrfs file store.

Type

Boolean

Required

No. Only used for btrfs.

Default

true

8.8. JOURNAL

filestore_journal_parallel

Description

Enables parallel journaling, default for btrfs.

Type

Boolean

Required

No

Default

false

filestore_journal_writeahead

Description

Enables write-ahead journaling, default for XFS.

Type

Boolean

Required

No

Default

false

filestore_journal_trailing

Description

Deprecated, never use.

Type

Boolean

Required

No

Default

false

8.9. MISCELLANEOUS

filestore_merge_threshold

Description

Minimum number of files in a subdirectory before merging into parent NOTE: A negative value means to disable subdirectory merging.

Type

Integer

Required

No

Default

10

filestore_split_multiple

Description

filestore_split_multiple * abs(filestore_merge_threshold) * 16 is the maximum number of files in a subdirectory before splitting into child directories.

Type

Integer

Required

No

Default

2

filestore_update_to

Description

Limits file store auto upgrade to specified version.

Type
: Integer

Required
: No

Default
: **1000**

filestore_blackhole

Description
: Drop any new transactions on the floor.

Type
: Boolean

Required
: No

Default
: **false**

filestore_dump_file

Description
: File onto which store transaction dumps.

Type
: Boolean

Required
: No

Default
: **false**

filestore_kill_at

Description
: Inject a failure at the n’th opportunity.

Type
: String

Required
: No

Default
: **false**

filestore_fail_eio

Description
: Fail or terminate unexpectedly on EIO.

Type
: Boolean

Required

No

Default

true

CHAPTER 9. JOURNAL CONFIGURATION REFERENCE

Ceph OSDs use a journal for the following reasons:

Speed

The journal enables the Ceph OSD Daemon to commit small write operations quickly. Ceph writes small, random I/O to the journal sequentially, which tends to speed up bursty workloads by allowing the backing file system more time to coalesce write operations. The Ceph OSD Daemon's journal, however, can lead to spiky performance with short spurts of high-speed writes followed by periods without any write progress as the file system catches up to the journal.

Consistency

Ceph OSD Daemons require a file system interface that guarantees atomic compound operations. Ceph OSD Daemons write a description of the operation to the journal and apply the operation to the file system. This enables atomic updates to an object (for example, placement group metadata). Every few seconds—between **filestore max sync interval** and **filestore min sync interval** settings—the Ceph OSD stops write operations and synchronizes the journal with the file system, allowing Ceph OSDs to trim operations from the journal and reuse the space. On failure, Ceph OSDs replay the journal starting after the last synchronization operation.

9.1. SETTINGS

Ceph OSD Daemons support the following journal settings:

journal_dio

Description

Enables direct I/O to the journal. Requires the **journal block align** option set to **true**.

Type

Boolean

Required

Yes when using **aio**.

Default

true

journal_aio

Description

Enables using **libaio** for asynchronous writes to the journal. Requires the **journal dio** option set to **true**.

Type

Boolean

Required

No.

Default

true.

journal_block_align

Description

Block aligns write operations. Required for **dio** and **aio**.

Type
: Boolean

Required
: Yes when using **dio** and **aio**.

Default
: **true**

journal_max_write_bytes

Description
: The maximum number of bytes the journal will write at any one time.

Type
: Integer

Required
: No

Default
: **10 << 20**

journal_max_write_entries

Description
: The maximum number of entries the journal will write at any one time.

Type
: Integer

Required
: No

Default
: **100**

journal_queue_max_ops

Description
: The maximum number of operations allowed in the queue at any one time.

Type
: Integer

Required
: No

Default
: **500**

journal_queue_max_bytes

Description
: The maximum number of bytes allowed in the queue at any one time.

Type
: Integer

Required

No

Default

10 << 20

journal_align_min_size

Description

Align data payloads greater than the specified minimum.

Type

Integer

Required

No

Default

64 << 10

journal_zero_on_create

Description

Causes the file store to overwrite the entire journal with **0's during `mkfs**.

Type

Boolean

Required

No

Default

false

CHAPTER 10. LOGGING CONFIGURATION REFERENCE

Logging and debugging settings are not required in a Ceph configuration file, but you can override default settings as needed.

The options take a single item that is assumed to be the default for all daemons regardless of channel. For example, specifying "info" is interpreted as "default=info". However, options can also take key/value pairs. For example, "default=daemon audit=local0" is interpreted as "default all to 'daemon', override 'audit' with 'local0'."

Ceph supports the following settings:

log_file

Description
The location of the logging file for the cluster.

Type
String

Required
No

Default
/var/log/ceph/$cluster-$name.log

mon_cluster_log_file

Description
The location of the monitor cluster's log file.

Type
String

Required
No

Default
/var/log/ceph/$cluster.log

log_max_new

Description
The maximum number of new log files.

Type
Integer

Required
No

Default
1000

log_max_recent

Description
The maximum number of recent events to include in a log file.

Type
: Integer

Required
: No

Default
: **1000000**

log_flush_on_exit

Description
: Determines if Ceph flushes the log files after exit.

Type
: Boolean

Required
: No

Default
: **true**

mon_cluster_log_file_level

Description
: The level of file logging for the monitor cluster. Valid settings include "debug", "info", "sec", "warn", and "error".

Type
: String

Default
: **"info"**

log_to_stderr

Description
: Determines if logging messages appear in **stderr**.

Type
: Boolean

Required
: No

Default
: **true**

err_to_stderr

Description
: Determines if error messages appear in **stderr**.

Type
: Boolean

Required
: No

Default
: **true**

log_to_syslog

Description
: Determines if logging messages appear in **syslog**.

Type
: Boolean

Required
: No

Default
: **false**

err_to_syslog

Description
: Determines if error messages appear in **syslog**.

Type
: Boolean

Required
: No

Default
: **false**

clog_to_syslog

Description
: Determines if **clog** messages will be sent to **syslog**.

Type
: Boolean

Required
: No

Default
: **false**

mon_cluster_log_to_syslog

Description
: Determines if the cluster log will be output to **syslog**.

Type
: Boolean

Required
: No

Default
: **false**

mon_cluster_log_to_syslog_level

Description

The level of syslog logging for the monitor cluster. Valid settings include "debug", "info", "sec", "warn", and "error".

Type

String

Default

"info"

mon_cluster_log_to_syslog_facility

Description

The facility generating the syslog output. This is usually set to "daemon" for the Ceph daemons.

Type

String

Default

"daemon"

clog_to_monitors

Description

Determines if **clog** messages will be sent to monitors.

Type

Boolean

Required

No

Default

true

mon_cluster_log_to_graylog

Description

Determines if the cluster will output log messages to graylog.

Type

String

Default

"false"

mon_cluster_log_to_graylog_host

Description

The IP address of the graylog host. If the graylog host is different from the monitor host, override this setting with the appropriate IP address.

Type

String

Default

"127.0.0.1"

mon_cluster_log_to_graylog_port

Description
: Graylog logs will be sent to this port. Ensure the port is open for receiving data.

Type
: String

Default
: **"12201"**

10.1. OSD

osd_preserve_trimmed_log

Description
: Preserves trimmed logs after trimming.

Type
: Boolean

Required
: No

Default
: **false**

osd_tmapput_sets_uses_tmap

Description
: Uses **tmap**. For debug only.

Type
: Boolean

Required
: No

Default
: **false**

osd_min_pg_log_entries

Description
: The minimum number of log entries for placement groups.

Type
: 32-bit Unsigned Integer

Required
: No

Default
: 1000

osd_op_log_threshold

Description

How many op log messages to show up in one pass.

Type

Integer

Required

No

Default

5

10.2. FILE STORE

filestore_debug_omap_check

Description

Debugging check on synchronization. This is an expensive operation.

Type

Boolean

Required

No

Default

0

10.3. THE CEPH OBJECT GATEWAY

rgw_log_nonexistent_bucket

Description

Log non-existent buckets.

Type

Boolean

Required

No

Default

false

rgw_log_object_name

Description

Log an object’s name.

Type

String

Required

No

Default

%Y-%m-%d-%H-%i-%n

rgw_log_object_name_utc

Description

Object log name contains UTC.

Type

Boolean

Required

No

Default

false

rgw_enable_ops_log

Description

Enables logging of every RGW operation.

Type

Boolean

Required

No

Default

true

rgw_enable_usage_log

Description

Enable logging of RGW’s bandwidth usage.

Type

Boolean

Required

No

Default

true

rgw_usage_log_flush_threshold

Description

Threshold to flush pending log data.

Type

Integer

Required

No

Default

1024

rgw_usage_log_tick_interval

Description

Flush pending log data every **s** seconds.

Type

Integer

Required

No

Default

30

rgw_intent_log_object_name

Description, Type

String

Required

No

Default

%Y-%m-%d-%i-%n

rgw_intent_log_object_name utc

Description

Include a UTC time stamp in the intent log object name.

Type

Boolean

Required

No

Default

false