HEALTH CHECKS

OVERVIEW

There is a finite set of possible health messages that a Ceph cluster can raise – these are defined as *health checks* which have unique identifiers.

The identifier is a terse pseudo-human-readable (i.e. like a variable name) string. It is intended to enable tools (such as UIs) to make sense of health checks, and present them in a way that reflects their meaning.

This page lists the health checks that are raised by the monitor and manager daemons. In addition to these, you may also see health checks that originate from MDS daemons (see CephFS health messages), and health checks that are defined by ceph-mgr python modules.

DEFINITIONS

MANAGER

MGR_MODULE_DEPENDENCY

An enabled manager module is failing its dependency check. This health check should come with an explanatory message from the module about the problem.

For example, a module might report that a required package is not installed: install the required package and restart your manager daemons.

This health check is only applied to enabled modules. If a module is not enabled, you can see whether it is reporting dependency issues in the output of *ceph module ls*.

MGR_MODULE_ERROR

A manager module has experienced an unexpected error. Typically, this means an unhandled exception was raised from the module's *serve* function. The human readable description of the error may be obscurely worded if the exception did not provide a useful description of itself.

This health check may indicate a bug: please open a Ceph bug report if you think you have encountered a bug.

If you believe the error is transient, you may restart your manager daemon(s), or use *ceph mgr fail* on the active daemon to prompt a failover to another daemon.

OSDS

OSD_DOWN

One or more OSDs are marked down. The ceph-osd daemon may have been stopped, or peer OSDs may be unable to reach the OSD over the network. Common causes include a stopped or crashed daemon, a down host, or a network outage.

Verify the host is healthy, the daemon is started, and network is functioning. If the daemon has crashed, the daemon log file (`/var/log/ceph/ceph-osd.*`) may contain debugging information.

OSD_<CRUSH TYPE>_DOWN

(e.g. OSD_HOST_DOWN, OSD_ROOT_DOWN)

All the OSDs within a particular CRUSH subtree are marked down, for example all OSDs on a host.

OSD_ORPHAN

An OSD is referenced in the CRUSH map hierarchy but does not exist.

The OSD can be removed from the CRUSH hierarchy with:

```
ceph osd crush rm osd.<id>
```

OSD_OUT_OF_ORDER_FULL

The utilization thresholds for *backfillfull*, *nearfull*, *full*, and/or *failsafe_full* are not ascending. In particular, we expect *backfillfull* < *nearfull*, *nearfull* < *full*, and *full* < *failsafe_full*.

The thresholds can be adjusted with:

```
ceph osd set-backfillfull-ratio <ratio>
ceph osd set-nearfull-ratio <ratio>
ceph osd set-full-ratio <ratio>
```

OSD_FULL

One or more OSDs has exceeded the *full* threshold and is preventing the cluster from servicing writes.

Utilization by pool can be checked with:

```
ceph df
```

The currently defined *full* ratio can be seen with:

```
ceph osd dump | grep full_ratio
```

A short-term workaround to restore write availability is to raise the full threshold by a small amount:

```
ceph osd set-full-ratio <ratio>
```

New storage should be added to the cluster by deploying more OSDs or existing data should be deleted in order to free up space.

OSD_BACKFILLFULL

One or more OSDs has exceeded the *backfillfull* threshold, which will prevent data from being allowed to rebalance to this device. This is an early warning that rebalancing may not be able to complete and that the cluster is approaching full.

Utilization by pool can be checked with:

```
ceph df
```

OSD_NEARFULL

One or more OSDs has exceeded the *nearfull* threshold. This is an early warning that the cluster is approaching full.

Utilization by pool can be checked with:

```
ceph df
```

OSDMAP_FLAGS

One or more cluster flags of interest has been set. These flags include:

- *full* - the cluster is flagged as full and cannot service writes
- *pauserd*, *pausewr* - paused reads or writes
- *noup* - OSDs are not allowed to start
- *nodown* - OSD failure reports are being ignored, such that the monitors will not mark OSDs *down*
- *noin* - OSDs that were previously marked *out* will not be marked back *in* when they start
- *noout* - down OSDs will not automatically be marked out after the configured interval
- *nobackfill*, *norecover*, *norebalance* - recovery or data rebalancing is suspended
- *noscrub*, *nodeep_scrub* - scrubbing is disabled
- *notieragent* - cache tiering activity is suspended

With the exception of *full*, these flags can be set or cleared with:

```
ceph osd set <flag>
ceph osd unset <flag>
```

OSD_FLAGS

One or more OSDs has a per-OSD flag of interest set. These flags include:

- *noup*: OSD is not allowed to start
- *nodown*: failure reports for this OSD will be ignored
- *noin*: if this OSD was previously marked *out* automatically after a failure, it will not be marked in when it stats
- *noout*: if this OSD is down it will not automatically be marked *out* after the configured interval

Per-OSD flags can be set and cleared with:

```
ceph osd add-<flag> <osd-id>
ceph osd rm-<flag> <osd-id>
```

For example,

```
ceph osd rm-nodown osd.123
```

OLD_CRUSH_TUNABLES

The CRUSH map is using very old settings and should be updated. The oldest tunables that can be used (i.e., the oldest client version that can connect to the cluster) without triggering this health warning is determined by the `mon_crush_min_required_version` config option. See Tunables for more information.

OLD_CRUSH_STRAW_CALC_VERSION

The CRUSH map is using an older, non-optimal method for calculating intermediate weight values for `straw` buckets.

The CRUSH map should be updated to use the newer method (`straw_calc_version=1`). See Tunables for more information.

CACHE_POOL_NO_HIT_SET

One or more cache pools is not configured with a *hit set* to track utilization, which will prevent the tiering agent from identifying cold objects to flush and evict from the cache.

Hit sets can be configured on the cache pool with:

```
ceph osd pool set <poolname> hit_set_type <type>
ceph osd pool set <poolname> hit_set_period <period-in-seconds>
ceph osd pool set <poolname> hit_set_count <number-of-hitsets>
ceph osd pool set <poolname> hit_set_fpp <target-false-positive-rate>
```

OSD_NO_SORTBITWISE

No pre-luminous v12.y.z OSDs are running but the `sortbitwise` flag has not been set.

The `sortbitwise` flag must be set before luminous v12.y.z or newer OSDs can start. You can safely set the flag with:

```
ceph osd set sortbitwise
```

POOL_FULL

One or more pools has reached its quota and is no longer allowing writes.

Pool quotas and utilization can be seen with:

```
ceph df detail
```

You can either raise the pool quota with:

```
ceph osd pool set-quota <poolname> max_objects <num-objects>
ceph osd pool set-quota <poolname> max_bytes <num-bytes>
```

or delete some existing data to reduce utilization.

DATA HEALTH (POOLS & PLACEMENT GROUPS)

PG_AVAILABILITY

Data availability is reduced, meaning that the cluster is unable to service potential read or write requests for some data in the cluster. Specifically, one or more PGs is in a state that does not allow IO requests to be serviced. Problematic PG states include *peering*, *stale*, *incomplete*, and the lack of *active* (if those conditions do not clear quickly).

Detailed information about which PGs are affected is available from:

```
ceph health detail
```

In most cases the root cause is that one or more OSDs is currently down; see the dicussion for `OSD_DOWN` above.

The state of specific problematic PGs can be queried with:

```
ceph tell <pgid> query
```

PG_DEGRADED

Data redundancy is reduced for some data, meaning the cluster does not have the desired number of replicas for all data (for replicated pools) or erasure code fragments (for erasure coded pools). Specifically, one or more PGs:

- has the *degraded* or *undersized* flag set, meaning there are not enough instances of that placement group in the cluster;
- has not had the *clean* flag set for some time.

Detailed information about which PGs are affected is available from:

```
ceph health detail
```

In most cases the root cause is that one or more OSDs is currently down; see the dicussion for `OSD_DOWN` above.

The state of specific problematic PGs can be queried with:

```
ceph tell <pgid> query
```

PG_DEGRADED_FULL

Data redundancy may be reduced or at risk for some data due to a lack of free space in the cluster. Specifically, one or more PGs has the *backfill_toofull* or *recovery_toofull* flag set, meaning that the cluster is unable to migrate or recover data because one or more OSDs is above the *backfillfull* threshold.

See the discussion for *OSD_BACKFILLFULL* or *OSD_FULL* above for steps to resolve this condition.

PG_DAMAGED

Data scrubbing has discovered some problems with data consistency in the cluster. Specifically, one or more PGs has the *inconsistent* or *snaptrim_error* flag is set, indicating an earlier scrub operation found a problem, or that the *repair* flag is set, meaning a repair for such an inconsistency is currently in progress.

See Repairing PG inconsistencies for more information.

OSD_SCRUB_ERRORS

Recent OSD scrubs have uncovered inconsistencies. This error is generally paired with *PG_DAMANGED* (see above).

See Repairing PG inconsistencies for more information.

CACHE_POOL_NEAR_FULL

A cache tier pool is nearly full. Full in this context is determined by the `target_max_bytes` and `target_max_objects` properties on the cache pool. Once the pool reaches the target threshold, write requests to the pool may block while data is flushed and evicted from the cache, a state that normally leads to very high latencies and poor performance.

The cache pool target size can be adjusted with:

```
ceph osd pool set <cache-pool-name> target_max_bytes <bytes>
ceph osd pool set <cache-pool-name> target_max_objects <objects>
```

Normal cache flush and evict activity may also be throttled due to reduced availability or performance of the base tier, or overall cluster load.

TOO_FEW_PGS

The number of PGs in use in the cluster is below the configurable threshold of `mon_pg_warn_min_per_osd` PGs per OSD. This can lead to suboptimizal distribution and balance of data across the OSDs in the cluster, and similar reduce overall performance.

This may be an expected condition if data pools have not yet been created.

The PG count for existing pools can be increased or new pools can be created. Please refer to Choosing the number of Placement Groups for more information.

TOO_MANY_PGS

The number of PGs in use in the cluster is above the configurable threshold of `mon_max_pg_per_osd` PGs per OSD. If this threshold is exceed the cluster will not allow new pools to be created, pool *pg_num* to be increased, or pool replication to be increased (any of which would lead to more PGs in the cluster). A large number of PGs can lead to higher memory utilization for OSD daemons, slower peering after cluster state changes (like OSD restarts, additions, or removals), and higher load on the Manager and Monitor daemons.

The simplest way to mitigate the problem is to increase the number of OSDs in the cluster by adding more hardware. Note that the OSD count used for the purposes of this health check is the number of “in” OSDs, so marking “out” OSDs “in” (if there are any) can also help:

```
ceph osd in <osd id(s)>
```

Please refer to Choosing the number of Placement Groups for more information.

SMALLER_PGP_NUM

One or more pools has a pgp_num value less than pg_num. This is normally an indication that the PG count was increased without also increasing the placement behavior.

This is sometimes done deliberately to separate out the *split* step when the PG count is adjusted from the data migration that is needed when pgp_num is changed.

This is normally resolved by setting pgp_num to match pg_num, triggering the data migration, with:

```
ceph osd pool set <pool> pgp_num <pg-num-value>
```

MANY_OBJECTS_PER_PG

One or more pools has an average number of objects per PG that is significantly higher than the overall cluster average. The specific threshold is controlled by the mon_pg_warn_max_object_skew configuration value.

This is usually an indication that the pool(s) containing most of the data in the cluster have too few PGs, and/or that other pools that do not contain as much data have too many PGs. See the discussion of *TOO_MANY_PGS* above.

The threshold can be raised to silence the health warning by adjusting the mon_pg_warn_max_object_skew config option on the monitors.

POOL_APP_NOT_ENABLED

A pool exists that contains one or more objects but has not been tagged for use by a particular application.

Resolve this warning by labeling the pool for use by an application. For example, if the pool is used by RBD,:

```
rbd pool init <poolname>
```

If the pool is being used by a custom application 'foo', you can also label via the low-level command:

```
ceph osd pool application enable foo
```

For more information, see Associate Pool to Application.

POOL_FULL

One or more pools has reached (or is very close to reaching) its quota. The threshold to trigger this error condition is controlled by the mon_pool_quota_crit_threshold configuration option.

Pool quotas can be adjusted up or down (or removed) with:

```
ceph osd pool set-quota <pool> max_bytes <bytes>
ceph osd pool set-quota <pool> max_objects <objects>
```

Setting the quota value to 0 will disable the quota.

POOL_NEAR_FULL

One or more pools is approaching is quota. The threshold to trigger this warning condition is controlled by the mon_pool_quota_warn_threshold configuration option.

Pool quotas can be adjusted up or down (or removed) with:

```
ceph osd pool set-quota <pool> max_bytes <bytes>
ceph osd pool set-quota <pool> max_objects <objects>
```

Setting the quota value to 0 will disable the quota.

OBJECT_MISPLACED

One or more objects in the cluster is not stored on the node the cluster would like it to be stored on. This is an indication that data migration due to some recent cluster change has not yet completed.

Misplaced data is not a dangerous condition in and of itself; data consistency is never at risk, and old copies of objects are never removed until the desired number of new copies (in the desired locations) are present.

OBJECT_UNFOUND

One or more objects in the cluster cannot be found. Specifically, the OSDs know that a new or updated copy of an object should exist, but a copy of that version of the object has not been found on OSDs that are currently online.

Read or write requests to unfound objects will block.

Ideally, a down OSD can be brought back online that has the more recent copy of the unfound object. Candidate OSDs can be identified from the peering state for the PG(s) responsible for the unfound object:

```
ceph tell <pgid> query
```

If the latest copy of the object is not available, the cluster can be told to roll back to a previous version of the object. See Unfound Objects for more information.

SLOW_OPS

One or more OSD requests is taking a long time to process. This can be an indication of extreme load, a slow storage device, or a software bug.

The request queue on the OSD(s) in question can be queried with the following command, executed from the OSD host:

```
ceph daemon osd.<id> ops
```

A summary of the slowest recent requests can be seen with:

```
ceph daemon osd.<id> dump_historic_ops
```

The location of an OSD can be found with:

```
ceph osd find osd.<id>
```

PG_NOT_SCRUBBED

One or more PGs has not been scrubbed recently. PGs are normally scrubbed every `mon_scrub_interval` seconds, and this warning triggers when `mon_warn_not_scrubbed` such intervals have elapsed without a scrub.

PGs will not scrub if they are not flagged as *clean*, which may happen if they are misplaced or degraded (see *PG_AVAILABILITY* and *PG_DEGRADED* above).

You can manually initiate a scrub of a clean PG with:

```
ceph pg scrub <pgid>
```

PG_NOT_DEEP_SCRUBBED

One or more PGs has not been deep scrubbed recently. PGs are normally scrubbed every `osd_deep_mon_scrub_interval` seconds, and this warning triggers when `mon_warn_not_deep_scrubbed` such intervals have elapsed without a scrub.

PGs will not (deep) scrub if they are not flagged as *clean*, which may happen if they are misplaced or degraded (see *PG_AVAILABILITY* and *PG_DEGRADED* above).

You can manually initiate a scrub of a clean PG with:

```
ceph pg deep-scrub <pgid>
```