TROUBLESHOOTING PGS

PLACEMENT GROUPS NEVER GET CLEAN

There are a few cases where Ceph placement groups never get clean:

1. **One OSD:** If you deviate from the quick start and use only one OSD, you will likely run into problems. OSDs report other OSDs to the monitor, and also interact with other OSDs when replicating data. If you have only one OSD, a second OSD cannot check its heartbeat. Also, if you remove an OSD and have only one OSD remaining, you may encounter problems. An secondary or tertiary OSD expects another OSD to tell it which placement groups it should have. The lack of another OSD prevents this from occurring. So a placement group can remain stuck “stale” forever.
2. **Pool Size = 1**: If you have only one copy of an object, no other OSD will tell the OSD which objects it should have. For each placement group mapped to the remaining OSD (see `ceph pg dump`), you can force the OSD to notice the placement groups it needs by running:

   ```
   ceph pg force_create_pg <pgid>
   ```

   1. **CRUSH Rules:** Another candidate for placement groups remaining unclean involves errors in your CRUSH map.

As a general rule, you should run your cluster with more than one OSD and a pool size greater than 1 object replica.

STUCK PLACEMENT GROUPS

It is normal for placement groups to enter states like “degraded” or “peering” following a failure. Normally these states indicate the normal progression through the failure recovery process. However, if a placement group stays in one of these states for a long time this may be an indication of a larger problem. For this reason, the monitor will warn when placement groups get “stuck” in a non-optimal state. Specifically, we check for:

- `inactive` - The placement group has not been `active` for too long (i.e., it hasn’t been able to service read/write requests).
- `unclean` - The placement group has not been `clean` for too long (i.e., it hasn’t been able to completely recover from a previous failure).
- `stale` - The placement group status has not been updated by a `ceph-osd`, indicating that all nodes storing this placement group may be down.

You can explicitly list stuck placement groups with one of:

```
ceph pg dump_stuck stale
ceph pg dump_stuck inactive
ceph pg dump_stuck unclean
```

For stuck `stale` placement groups, it is normally a matter of getting the right `ceph-osd` daemons running again. For stuck `inactive` placement groups, it is usually a peering problem (see Placement Group Down - Peering Failure). For stuck `unclean` placement groups, there is usually something preventing recovery from completing, like unfound objects (see Unfound Objects);

PLACEMENT GROUP DOWN - PEERING FAILURE

In certain cases, the `ceph-osd` *Peering* process can run into problems, preventing a PG from becoming active and usable. For example, `ceph health` might report:

```
ceph health detail
HEALTH_ERR 7 pgs degraded; 12 pgs down; 12 pgs peering; 1 pgs recovering; 6 pgs stuck unclean
...
pg 0.5 is down+peering
pg 1.4 is down+peering
...
```

```
osd.1 is down since epoch 69, last address 192.168.106.220:6801/8651
```

We can query the cluster to determine exactly why the PG is marked down with:

```
ceph pg 0.5 query
```

```
{ "state": "down+peering",
  ...
  "recovery_state": [
        { "name": "Started\/Primary\/Peering\/GetInfo",
          "enter_time": "2012-03-06 14:40:16.169679",
          "requested_info_from": []},
        { "name": "Started\/Primary\/Peering",
          "enter_time": "2012-03-06 14:40:16.169659",
          "probing_osds": [
                0,
                1],
          "blocked": "peering is blocked due to down osds",
          "down_osds_we_would_probe": [
                1],
          "peering_blocked_by": [
                { "osd": 1,
                  "current_lost_at": 0,
                  "comment": "starting or marking this osd lost may let us proceed"}]},
        { "name": "Started",
          "enter_time": "2012-03-06 14:40:16.169513"}
    ]
}
```

The `recovery_state` section tells us that peering is blocked due to down `ceph-osd` daemons, specifically `osd.1`. In this case, we can start that `ceph-osd` and things will recover.

Alternatively, if there is a catastrophic failure of `osd.1` (e.g., disk failure), we can tell the cluster that it is `lost` and to cope as best it can.

Important: This is dangerous in that the cluster cannot guarantee that the other copies of the data are consistent and up to date.

To instruct Ceph to continue anyway:

```
ceph osd lost 1
```

Recovery will proceed.

UNFOUND OBJECTS

Under certain combinations of failures Ceph may complain about `unfound` objects:

```
ceph health detail
HEALTH_WARN 1 pgs degraded; 78/3778 unfound (2.065%)
pg 2.4 is active+degraded, 78 unfound
```

This means that the storage cluster knows that some objects (or newer copies of existing objects) exist, but it hasn't found copies of them. One example of how this might come about for a PG whose data is on ceph-osds 1 and 2:

- 1 goes down
- 2 handles some writes, alone
- 1 comes up
- 1 and 2 repeer, and the objects missing on 1 are queued for recovery.
- Before the new objects are copied, 2 goes down.

Now 1 knows that these object exist, but there is no live `ceph-osd` who has a copy. In this case, IO to those objects will block,

and the cluster will hope that the failed node comes back soon; this is assumed to be preferable to returning an IO error to the user.

First, you can identify which objects are unfound with:

```
ceph pg 2.4 list_missing [starting offset, in json]
```

```
{ "offset": { "oid": "",
     "key": "",
     "snapid": 0,
     "hash": 0,
     "max": 0},
 "num_missing": 0,
 "num_unfound": 0,
 "objects": [
    { "oid": "object 1",
      "key": "",
      "hash": 0,
      "max": 0 },
    ...
 ],
 "more": 0}
```

If there are too many objects to list in a single result, the `more` field will be true and you can query for more. (Eventually the command line tool will hide this from you, but not yet.)

Second, you can identify which OSDs have been probed or might contain data:

```
ceph pg 2.4 query
```

```
"recovery_state": [
     { "name": "Started\/Primary\/Active",
       "enter_time": "2012-03-06 15:15:46.713212",
       "might_have_unfound": [
             { "osd": 1,
               "status": "osd is down"}]},
```

In this case, for example, the cluster knows that `osd.1` might have data, but it is down. The full range of possible states include:

```
* already probed
* querying
* osd is down
* not queried (yet)
```

Sometimes it simply takes some time for the cluster to query possible locations.

It is possible that there are other locations where the object can exist that are not listed. For example, if a ceph-osd is stopped and taken out of the cluster, the cluster fully recovers, and due to some future set of failures ends up with an unfound object, it won't consider the long-departed ceph-osd as a potential location to consider. (This scenario, however, is unlikely.)

If all possible locations have been queried and objects are still lost, you may have to give up on the lost objects. This, again, is possible given unusual combinations of failures that allow the cluster to learn about writes that were performed before the writes themselves are recovered. To mark the "unfound" objects as "lost":

```
ceph pg 2.5 mark_unfound_lost revert
```

This the final argument specifies how the cluster should deal with lost objects. Currently the only supported option is "revert", which will either roll back to a previous version of the object or (if it was a new object) forget about it entirely. Use this with caution, as it may confuse applications that expected the object to exist.

HOMELESS PLACEMENT GROUPS

It is possible for all OSDs that had copies of a given placement groups to fail. If that's the case, that subset of the object store is unavailable, and the monitor will receive no status updates for those placement groups. To detect this situation, the monitor marks any placement group whose primary OSD has failed as `stale`. For example:

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

You can identify which placement groups are `stale`, and what the last OSDs to store them were, with:

```
ceph health detail
HEALTH_WARN 24 pgs stale; 3/300 in osds are down
...
pg 2.5 is stuck stale+active+remapped, last acting [2,0]
...
osd.10 is down since epoch 23, last address 192.168.106.220:6800/11080
osd.11 is down since epoch 13, last address 192.168.106.220:6803/11539
osd.12 is down since epoch 24, last address 192.168.106.220:6806/11861
```

If we want to get placement group 2.5 back online, for example, this tells us that it was last managed by `osd.0` and `osd.2`. Restarting those `ceph-osd` daemons will allow the cluster to recover that placement group (and, presumably, many others).

ONLY A FEW OSDS RECEIVE DATA

If you have many nodes in your cluster and only a few of them receive data, check the number of placement groups in your pool. Since placement groups get mapped to OSDs, a small number of placement groups will not distribute across your cluster. Try creating a pool with a placement group count that is a multiple of the number of OSDs. See Placement Groups for details. The default placement group count for pools isn't useful, but you can change it here.

CAN'T WRITE DATA

If your cluster is up, but some OSDs are down and you cannot write data, check to ensure that you have the minimum number of OSDs running for the placement group. If you don't have the minimum number of OSDs running, Ceph will not allow you to write data because there is no guarantee that Ceph can replicate your data. See `osd pool default min size` in the Pool, PG and CRUSH Config Reference for details.