```
pool 1 'device_health_metrics' replicated size 3 min_size 2 crush_rule 0
 object_hash rjenkins pg_num 1 pgp_num 1 autoscale_mode on last_change 475 flags
 hashpspool stripe_width 0 pg_num_min 1 application mgr_devicehealth
...output omitted...
osd.0 up   in  weight 1 up_from 471 up_thru 471 down_at 470 last_clean_interval
 [457,466) [v2:172.25.250.12:6801/1228351148,v1:172.25.250.12:6802/1228351148]
 [v2:172.25.249.12:6803/1228351148,v1:172.25.249.12:6804/1228351148] exists,up
 cfe311b0-dea9-4c0c-a1ea-42aaac4cb160
...output omitted...
```

Analyzing OSD Map Updates

Ceph updates the OSD map every time an OSD joins or leaves the cluster. An OSD can leave the Ceph cluster either because of an OSD failure or a hardware failure.

Even though the cluster map as a whole is maintained by the MONs, OSDs do not use a leader to manage the OSD map; they propagate the map among themselves. OSDs tag every message they exchange with the OSD map epoch. When an OSD detects that it is lagging behind, it performs a map update with its peer OSD.

In large clusters, where OSD map updates are frequent, it is not practical to always distribute the full map. Instead, receiving OSD nodes perform incremental map updates.

Ceph also tags the messages between OSDs and clients with the epoch. Whenever a client connects to an OSD, the OSD inspects the epoch. If the epoch does not match, then the OSD responds with the correct increment so that the client can update its OSD map. This negates the need for aggressive propagation, because clients learn about the updated map only at the time of next contact.

Updating Cluster Maps with Paxos

To access a Ceph cluster, a client first retrieves a copy of the cluster map from the MONs. All MONs must have the same cluster map for the cluster to function correctly.

MONs use the Paxos algorithm as a mechanism to ensure that they agree on the cluster state. Paxos is a distributed consensus algorithm. Every time a MON modifies a map, it sends the update to the other monitors through Paxos. Ceph only commits the new version of the map after a majority of monitors agree on the update.

The MON submits a map update to Paxos and only writes the new version to the local key-value store after Paxos acknowledges the update. The read operations directly access the key-value store.