Ceph Command Cheatsheet

General access

'ceph -s', 'ceph health {detail}' Display cluster status; add watch cmd for real time
'ceph -w' Display running cluster status
'**ceph report'** Generate a comprehensive report of ceph conf
'ceph status –f json | jq' Output ceph status in json format
'**ceph --watch-debug|watch-info|watch-sec|watch-warn|watch-error'** watch debug/ info/ security / warning /error events
'ceph df' show cluster utilization statistics
'ceph fs ls' List file systems; other cmd "new|reset|rm"
'rados df {detail}' Get rados statistics
'ceph node ls' list basic info of nodes (json)
'**ceph {-a} start|stop [mon|osd|mds|ceph-rados]'** start or stop ceph services; '-a': {daemon_type}.{id}; similar usage: 'service ceph –a start|stop
'**ceph daemon {daemon_type}.{id} config show'** Show config info (from the corresponding node); use "get|set config_parameter" see also "—admin-daemon"
'**ceph tell {daemon-type}.{id or *} injectargs --{config_parameter} {value}'** similar to above cmd; but from any node
'ceph version' display version info

OSD access

'ceph osd create' Create OSDs
'ceph auth del osd.x', 'ceph osd crush rm osd.x', 'ceph osd rm osd.x' Remove OSDs (**x=OSD ID**)
'ceph osd stat' see if all OSDs are running
'ceph osd tree' View the CRUSH map
'ceph osd repair x' Repair OSD x
'ceph osd scrub x' send a scrub cmd to OSD x
'ceph osd crush reweight {name} {weight}' Adjust an OSD's crush weight
'ceph osd crush remove {name}' remove OSD or bucket
'[start | stop] ceph-osd id=x' Start or stop a osd with ID x from out/down (not always working)
'ceph osd [out|in|down|lost] {x}' Mark OSD in/out/down
'ceph osd map {poolname} {object-name}' Find the object location
'**ceph osd find x'** Search for an OSD and its location in a crush map
'ceph osd getmap -o file' Write a copy of the most recent OSD map to a file
'ceph osd getcrushmap -o file' Write a copy of the crush map from the most recent OSD map to file
'ceph osd dump' Dump the OSD map; with grep to get more information, e.g., 'replicated size' to display per pool placement group and replication levels
'ceph osd map <pool-name> <object-name>' Find out where a specific object is or would be stored in system
'ceph osd pause| unpause' Pause or unpause IO to OSD
'ceph osd [unset | set] noout' Set or unset the osd for noout state for maintenance purpose

PG access

'ceph pg dump { -o {filename} --format=json |plain}' Check placement group stats
'ceph pg {poolnum}.y query' Query a particular placement group (y=PG ID)
'ceph pg dump_stuck [unclean | inactive | stale | undersized |degraded]' Identify stuck placement groups
'ceph pg map {poolnum}.y' Find OSD for a particular pg
'ceph pg scrub y' scrub a placement group
'ceph pg repair' {PG-ID} Repair PG-ID

Pool access

'ceph df' Display pool usage stats
'rados -p pool_name ls' List pieces of pool using rados & verify that object stored the object
'ceph osd lspools' List pools
'ceph osd pool create' ,'ceph osd pool delete' Create or delete a storage pool
'ceph osd pool create {pool-name} {pg-num} [{pgp-num}] [replicated] [crush-ruleset-name] [expected-num-objects]
ceph osd pool create {pool-name} {pg-num} {pgp-num} erasure [erasure-code-profile] [crush-ruleset-name] [expected_num_objects]'
'ceph osd pool create ecpool 12 12 erasure' Create an EC pool named ecpool with pg=pgp=12
'ceph osd pool set-quota {pool-name} [max_objects {obj-count}] [max_bytes {bytes}]' Set pool quotas for the maximum number of bytes and/or the maximum number of objects per pool
'ceph osd pool rename {current-pool-name} {new-pool-name}' Rename a pool
'ceph osd pool mksnap| rmsnap {pool-name} {snap-name}' make or remove a snapshot
'ceph osd pool set {pool-name} {key} {value}' Set pool values [size, min_size, crush_ruleset etc], e.g.,
'ceph osd pool set repool size 3' Set replicas num as 3
'ceph osd pool set pool_name pg_num 512' 'ceph osd pool set pool_name pgp_num 512' Set pool placement group sizes
'ceph osd pool get {pool-name} {key}' Get pool values
'ceph osd pool get poolname pg_num' Find out total number of placement groups being used by pool
'ceph osd dump | grep -i poolname' Find out replication level being used by pool
'**ceph osd pool delete ecpool ecpool --yes-i-really-really-mean-it**' delete ecpool
'rados mkpool <Name>; rados rmpool <Name>' Add or remove a pool to the configuration
'ceph osd blacklist ls' Display blacklisted clients

RBD access

'rbd -p pool_name list' Display rbd images in a pool:
'rbd --pool rbd snap create pool_name/image_name@snap_name' Create rbd snapshot
'rbd map {image}' Map image to OS; after mount, able to use dd; together with **'rbd resize'**
'rbd snap ls pool_name/image_name' Display rbd snapshots; together with **'snap create'**
'rbd showmapped' Display which images are mapped via kernel

jun.xu@wdc.com; version 0.6; updated 04 Jan 2017

Ceph Command Cheatsheet

'rbd create {image-name} --size {megabytes}' Create a block device
'rbd ls' See the block device by listing them
'**rbd info –image image-name'** Show image information
'**rbd map block-device-name'** Show map info of block device

MDS access

'ceph tell mds.{mds-id} injectargs --{switch} {value} [--{switch} {value}]' Change configuration parameters on a running mds.
'ceph mds stat' Enable debug messages
'ceph fs ls' Check the CephFS filesystem list
'ceph mds add_data_pool <pool>' Add data pool
'ceph mds cluster_down|cluster_up' take down or up cluster

Mon access

'ceph mon stat' Show monitor stats
'ceph mon dump' Dump monitor state
'ceph [-m monhost] {command}' Issue command to monitor, e.g., mon_status
'ceph quorum_status -f json-pretty' check mon quorum status for troubleshooting
'ceph compact' Compact monitor's leveldb storage
'ceph injectargs <injected_args> [<injected_args>...]' Inject conf args into monitor

CRUSH access

'ceph osd crush dump' View the crush map
'ceph osd crush rule list' View the crush map rules
'ceph osd crush rule dump <crush_rule_name>' View the detailed crush rule
'ceph osd getcrushmap -o crushmapdump' Obtain crush map
'crushtool -d crushmapdump -o crushmapdump-decompiled' Decompile crush map
'crushtool -c crushmapdump-decompiled -o crushmapdump-compiled ceph osd' Compile crush map
'setcrushmap -i crushmapdump-compiled' Set crush map

EC profile

**'ceph osd erasure-code-profile set {name} **
**[{directory=directory}] **
**[{plugin=plugin}] \ [ruleset-root={root}] **
**[ruleset-failure-domain={bucket-type}] **
[{key=value} ...] [--force]' Set EC profile
**'…\ plugin=jerasure k={data-chunks} m={coding-chunks} **
technique={reed_sol_van|reed_sol_r6_op|cauchy_orig|cauchy_good|liberation|blaum_roth|liber8tion} \ …',
**'…\ plugin=isa technique={reed_sol_van|cauchy} **
[k={data-chunks}] [m={coding-chunks}] \…'
'…\ **plugin=lrc k={data-chunks} m={coding-chunks} **
l={locality} [ruleset-locality={bucket-type}] \...'
**'…\ plugin=shec [k={data-chunks}] [m={coding-chunks}] **
[c={durability-estimator}] \...' 4 types of plug-ins
'ceph osd erasure-code-profile [get | rm] myprofile' Get or remove a EC profile; similar cmd **'ls'**

Benchmark access

'ceph tell osd.x bench [NUMER_OF_OBJECTS] [BYTES_PER_WRITE]' Bench write a OSD
'ceph osd perf' Display OSD commit and apply latency
'rados bench -p {pool-name} {seconds} [write | seq | rand] --no-cleanup' Beachmark an EC pool named ecpool with 10 seconds of write access without clearup [-t 100 (100 threads; default 16); –b size]; similar for **'fio'** without mount
'rados -p pool-name cleanup --prefix benchmark' Cleanup the pool
'**rbd bench-write block-device-name --io-total|io-threads|io-total|io-pattern <seq|rand>'** Bench write a block device mounted
**'rados -p rbd load-gen **
--num-objects 50 --min-object-size 4M --max-object-size 4M \ --max-ops 16 --min-op-len 4M --max-op-len 4M \ --percent 5 --target-throughput 2000 --run-length 60' Generate (simulated heavy) load on a Ceph cluster
'fio --size=10G --ioengine=rbd --invalidate=0 --direct=1 --numjobs=10 --rw=write --name=fiojob --blocksize_range=4K-512k --iodepth=1 --pool=bench --rbdname=fio-test' fio RBD benchmark

Auth access

'ceph auth list | import' List or import cluster keys
'ceph auth add <entity> {<caps> [<caps>...]}' Add authentication info for a particular entity
'ceph auth del|get|get-key|export <entity> ' Delete, get or export key
'ceph auth get-or-create|get-or-create-key <entity> {<caps> [<caps>...]}' get or add authentication/key info for a particular entity
'ceph config-key del|exists|get|list|put' Access configuration key

jun.xu@wdc.com; version 0.6; updated 04 Jan 2017